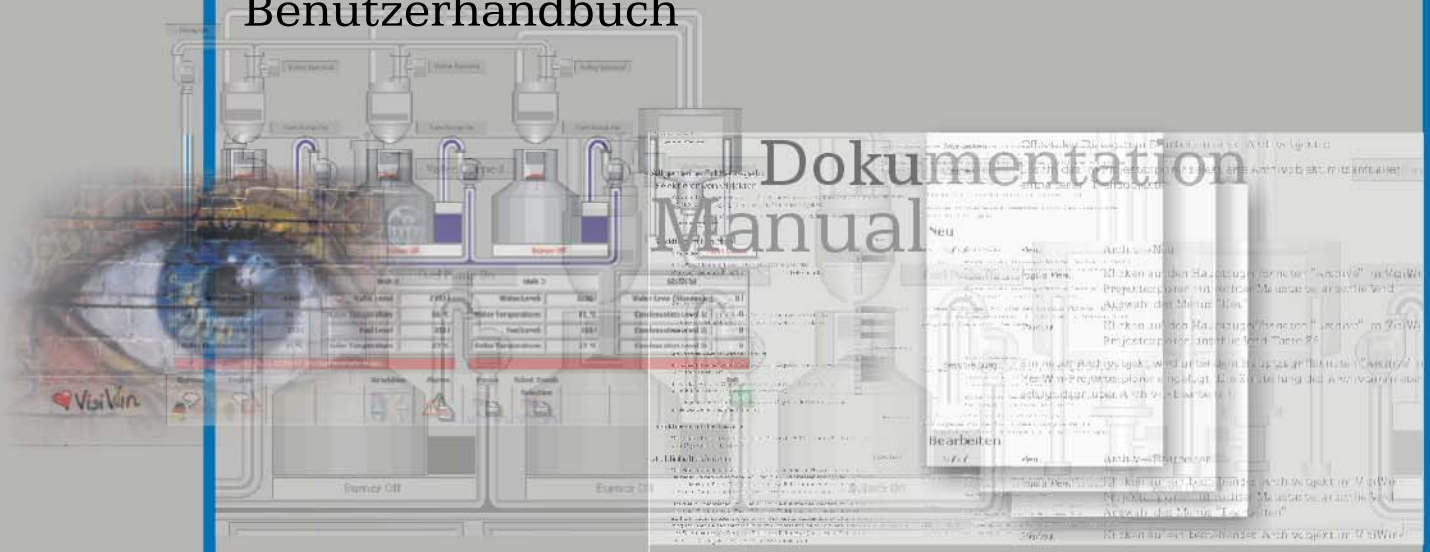
Automatisierungskonzepte

INOSOFT®

VisiWinNET 2005

Benutzerhandbuch



- VisiWin
  - VisiWinNET 2005
    - **Allgemeines**
    - Klassenbibliothek
    - Systeme
    - Werkzeuge
    - Technische Infos
  - Inosoft OPCServer
    - Grundlagen und Hilfsmittel
    - Protokolle

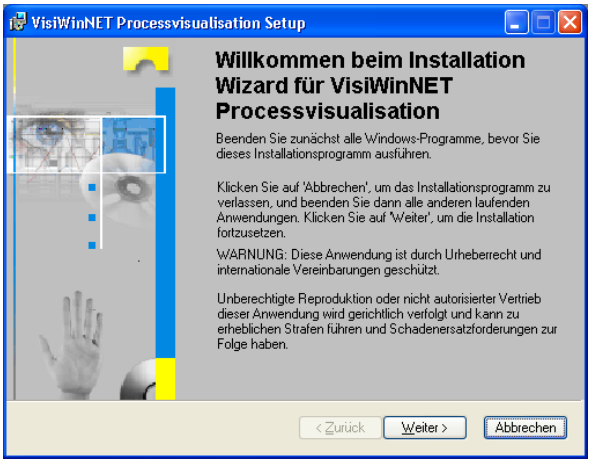
VisiWin

www.inosoft.com

Die Brücken zwischen Mensch und Maschine

VisiWinNET 2005

Benutzerhandbuch


VisiWinNET Processvisualisation Setup
Willkommen beim Installation Wizard für VisiWinNET Processvisualisation
Beenden Sie zunächst alle Windows-Programme, bevor Sie dieses Installationsprogramm ausführen.
Klicken Sie auf 'Abbrechen', um das Installationsprogramm zu verlassen, und beenden Sie dann alle anderen laufenden Anwendungen. Klicken Sie auf 'Weiter', um die Installation fortzusetzen.
WARNUNG: Diese Anwendung ist durch Urheberrecht und internationale Vereinbarungen geschützt.
Unberechtigte Reproduktion oder nicht autorisierter Vertrieb dieser Anwendung wird gerichtlich verfolgt und kann zu erheblichen Strafen führen und Schadenersatzforderungen zur Folge haben.
< Zurück
Weiter >
Abbrechen

VisiWinNET Installationsassistent

Ohne ausdrückliche schriftliche Erlaubnis der INOSOFT GmbH darf der Inhalt des Handbuches nicht anderweitig verwendet werden.
Den Inhalt des Handbuches haben wir auf Übereinstimmung mit der beschriebenen Software geprüft. Trotzdem können Abweichungen nicht ausgeschlossen werden. Aus diesem Grund können wir für eine völlige Übereinstimmung keine Gewähr geben. Der Inhalt des Handbuches wird regelmäßig auf Änderungen geprüft. Korrekturen sind in der nachfolgenden Ausgabe vorhanden. Verbessungsvorschläge werden dankbar angenommen.

Legende

Um auf besondere Einstellungen aufmerksam zu machen, werden in den INOSOFT-Dokumentationen folgende Symbole verwendet:

Achtung	Absätze mit diesem Zeichen sollten Sie besonders aufmerksam lesen - und beachten.
Hinweis	Wichtige Einstellung "additional informations".
Tipp	Viele Wege führen nach Rom, hier finden Sie eine Abkürzung.
In Arbeit	Funktionen in Vorbereitung, bzw. bereits implementiert, jedoch noch nicht für die Dokumentation aufbereitet.
Beispiel ausführen	Anweisungen, die in einem Beispiel auszuführen sind.
Beispiel beobachten	Ergebnisse, die durch die Ausführung der Beispielanweisungen zu beobachten sind.

© / ™ / ®
Windows®, Windows NT®, Windows 2000® , Windows XP®
sind eingetragene Warenzeichen der Firma Microsoft.
Weitere mit ® erwähnte Produktnamen sind Warenzeichen der betreffenden Herstellerfirmen.

INOSOFT GmbH

VisiWinNET Version: 6.04.000

erstellt am 07.06.2010

Inhalt

1 Vorwort 1

2 Willkommen 2

2.1 Installation von VisiWinNET Professional 5

2.1.1 Lieferumfang 5

2.1.2 Systemanforderungen Entwicklungssystem 5

2.1.3 Installationsanweisung 6

2.1.4 Softwareschutz 7

2.2 Schnelleinstieg 9

2.2.1 Steuerungsanbindung mit OPC 10

2.2.2 Steuerungsanbindung mit VisiWin-Treibern 15

2.2.3 Steuerungsanbindung mit VisiWinNET-Treibern 20

2.2.4 Darstellung und Manipulation von Prozesswerten 24

2.2.5 Grundregeln Applikationsaufbau 29

2.2.6 Sprachumschaltung 32

2.2.7 Alarmsystem 40

2.2.8 Archivsystem 50

2.2.9 Rezepte 52

2.2.10 Benutzerverwaltung 58

2.2.11 Protokollierung 64

3 Projekte unter Windows CE 65

3.1 Projektübertragung aus Visual Studio zum CE-Gerät 65

4 Weiterführende Referenzen 69

4.1 Hilfe zu den Systemen und Funktionen von VisiWinNET 69

4.1.1 Aufruf 70

4.1.2 Recherche 72

4.2 Hilfe zur Klassenbibliothek von VisiWinNET 73

4.2.1 Aufruf 73

4.2.2 Inhalt der Klassenbibliothekshilfe 76

4.3 Handbuch 80

4.3.1 PDF-Dateien 80

1 Vorwort

Zu diesem Handbuch

Dieses Handbuch enthält einleitende Informationen zu VisiWinNET Professional. Enthalten sind unter anderem Installationsanweisungen, Systemvoraussetzungen und ein erstes Beispiel, das den Umgang mit der Entwicklungsumgebung einführend erläutert.

Fragen und Probleme

Bei technischen Fragen und Problemen wenden Sie sich bitte an Ihre zuständige INOSOFT Vertretung oder an den Support der INOSOFT GmbH unter Tel.:+49 (5221) 16 66 02 bzw. E-Mail: Support@INOSOFT.com

Häufige Fragen und Probleme werden auch auf unserer Homepage unter www.inosoft.com behandelt.
Dort finden Sie auch einen Supportbereich, in dem Sie sich mit Fragen direkt an unsere Zentrale wenden können.

2 Willkommen

Willkommen bei VisiWinNET Professional, der Prozessvisualisierung unter Windows®.

VisiWinNET Professional ist ein Entwicklungspaket zur individuellen Gestaltung unterschiedlichster Prozessvisualisierungen. Dank des offenen und flexiblen Konzeptes können neben den herkömmlichen Standards ganz spezifische Sonderfunktionen implementiert werden.

Die Anwendungsbereiche von VisiWinNET decken die gesamte Palette der marktüblichen Anforderungen ab: vom CE-Kleinstterminal über die klassische Einzelplatzlösung bis hin zur komplexen Client-Serveranwendung oder SCADA/ Leitstandsapplikation ist alles möglich. Die unterschiedlichen Produktvarianten und die Staffelung der Laufzeitlizenzen (Anzahl von Prozesswerten) erlauben dabei die optimale Ausnutzung des Produktes.

VB.NET oder C#:

Eine Programmier-sprache?

VisiWinNET Professional ist ein Visualisierungssystem auf Basis der .NET Entwicklungsumgebung. Im wesentlichen werden hier die Programmiersprachen C# und VB.NET angeboten. Die Frage, wieso VisiWinNET Professional zur Funktion eine Programmiersprache benötigt, ist einfach zu beantworten:

Ein herkömmliches Visualisierungssystem ermöglicht die schnelle und komfortable Lösung von Standardaufgaben. Prozesswerte aus einer SPS sollen dargestellt, Eingaben an die SPS weitergeleitet werden. Zusätzliche Module ermöglichen weitere Funktionen, die sich historisch aus den Marktanforderungen der Visualisierungsbranche entwickelt haben, ohne die ein Visualisierungsprodukt beim Kunden also nicht bestehen kann.

VisiWinNET erfüllt diese Standards durch seinen modularen Aufbau. Der Variablenkernel dient als Verbindung zwischen der SPS und der Visualisierung. Die Zusatzmodule (Meldesystem, Wertaufzeichnung, Sprachumschaltung, ...) tauschen bei Bedarf mit dem Variablenkernel Daten aus und stellen sie der Visualisierungsoberfläche in aufbereiteter Form zur Verfügung. Die Darstellung dieser Informationen wird durch das für ein Visualisierungssystem gewohnt einfache "Platzieren und Parametrieren" von Steuerelementen projektiert.

Die Visualisierung auf der Basis einer Programmiersprache erweitert jedoch den Funktionsumfang erheblich:

- Der volle syntaktische Umfang einer modernen Programmiersprache steht dem Entwickler zur Verfügung, wenn eben nicht genau die Standardlösung gefordert ist.
- In Hinblick auf eine durchgängige Prozessauswertung (vertikale Integration) ist es wichtig, eine offene Schnittstelle zu andern Systemen verwenden zu können. C# und VB.NET verfolgen konsequent diesen Ansatz. "Remoting" und "COM" ermöglichen die Netzwerkfähigkeit, Datenbankzugriffe werden unterstützt.

Die Programmiersprache dient sozusagen als Schnittstelle zur kundenspezifischen Projektanpassung, die mit einem Standardprodukt entweder gar nicht oder nur durch kostenintensive Erweiterungen des Systemherstellers zu realisieren sind.

Arbeiten mit VisiWinNET Professional

Die von VisiWinNET zur Verfügung gestellten Komponenten sind auf den Datenaustausch mit Steuerungen spezialisiert. Sie ermöglichen den schnellen Aufbau einer Verbindung zwischen der Hardware und dem Visualisierungsprojekt. Übliche Erweiterungen, wie die Verwaltung von Meldungen, die Aufzeichnung von Variablenwerten sowie die Umschaltung der Applikationssprache, sind einfach und komfortabel zu projektieren. Die Erstellung einer Visualisierung verläuft im Wesentlichen in zwei Schritten:

Projektierung der Prozessdatenbank

Die zur Visualisierung benötigten Daten werden in den VisiWinNET-Editoren durch VisiWin-Definitionen bestimmt. Hier wird festgelegt, welche Zellen im SPS-Speicher für die Visualisierung benötigt werden und welche Sonderfunktionen (Alarm-/ Trendaufzeichnung...) sie erfüllen sollen.

Bedienoberfläche

Die Gestaltung der Bedienoberfläche wird durch die Projektierung von Formularen und Dialogen festgelegt. Die von VisiWinNET bereitgestellten Steuerelemente haben den vollen Zugriff auf die Prozessdatenbank. Variablen können angezeigt oder beschrieben werden. Die Sonderfunktionen Alarme und Trendaufzeichnungen werden durch spezielle Steuerelemente unterstützt. Zusätzlich unterstützen VisiWinNET-Steuerelemente die Anzeige von sprachumschaltbaren Texten. Damit kann eine international einsetzbare Applikation aufgebaut werden.

Neuerungen gegenüber VisiWinStudio

VisiWinNET ist die konsequente Weiterentwicklung des Vorgängerproduktes VisiWinStudio. Das als Basis verwendete .NET Framework bietet jedoch erweiterte Möglichkeiten:

- Kompatibilität: Microsoft verspricht durch die .NET-Technologie weitestgehende Plattformunabhängigkeit. Für unterschiedliche Plattformen steht jeweils ein Framework zur Verfügung, dass zu Programmen eine uniforme Schnittstelle aufweist und die Anpassungen zu Betriebssystem und Hardware regelt. Damit können erstmals auch Windows®-CE Geräte quasi nahtlos kompatibel mit den gleichen Entwicklungswerkzeugen behandelt werden, wie ein Standard-PC.
- Wahlfreiheit: Neben VB steht nun auch die Programmiersprache C# zur Verfügung.
- Versionsunabhängigkeit: .NET räumt mit den Versionskonflikten der Vergangenheit rigoros auf. Von unterschiedlichen Applikationen genutzte Komponenten können nun wieder im Applikationsverzeichnis installiert werden. Die konfliktbehaftete Komponentenregistrierung verliert dadurch an Bedeutung. Einen gemeinsames Systemverzeichnis existiert weiter im "Global Assembly Cache", birgt jedoch durch das verbesserte Versionsmanagement wesentlich weniger Gefahren: Hier können Dateien gleicher Identität mit unterschiedlichen Versionen parallel installiert werden. Eine Applikation ist nun wesentlich enger an die entsprechende Version einer Komponente gebunden als früher.

VisiWinNET wurde von Anfang an so konzeptioniert, dass es die oben genannten Vorteile unterstützt:

- Die Systeme und Bestandteile von VisiWinNET sind in den einzelnen Produktvarianten (Enterprise/Standard/Compact) weitestgehend funktional und bedientechnisch identisch.
- C# und VB.NET werden als Programmiersprachen unterstützt
- In Zukunft folgende Versionen mit erweitertem funktionalen Umfang können parallel zu bestehenden Versionen installiert werden. Insbesondere bei einer längerfristigen Entwicklung kann nun der Projekteur entscheiden, ob er ein Update wegen neuer Features wagt oder ob der aktuelle Projektstand erhalten bleiben soll.

2.1 Installation von VisiWinNET Professional

2.1.1 Lieferumfang

Der Lieferumfang des Programmpaketes VisiWinNET® Professional umfasst:

- CD VisiWinNET Version 6.xx
- Software-Schutz (Entwicklungsdongle, nur in der lizenzierten Version)

2.1.2 Systemanforderungen Entwicklungssystem

Computer	IBM kompatibler PC
CPU	Intel Pentium/Celeron Familie oder vergleichbare CPU ab 500 MHz Taktfrequenz; 1GHz oder mehr empfohlen.
Arbeitsspeicher	Mindestens 512MB oder mehr empfohlen.
Grafik	Mindestens 800x600 Pixel, High Color (16 Bit); 1280x1024 Pixel, True Color (32 Bit) oder mehr empfohlen.
Festplatte	300 MB freier Speicherplatz; zusätzlich 280 MB freier Speicherplatz, falls das .NET Framework 2.0 noch nicht installiert ist.
CD-ROM Laufwerk	Nur zur Installation
Schnittstellen	Für CE-Projekte: Ethernet Netzwerkkarte zur Kommunikation mit dem Zielgerät; alternativ freier serieller Port oder USB Anschluss. Zusätzlich ein freier paralleler Port oder USB-Anschluss für die Lizenzierung von VisiWinNET über einen Hardware-Dongle.
Betriebssystem	Microsoft Windows 2003 Server/XP SP2/Vista.
Software	Microsoft Visual Studio 2005 (ab Visual Basic/C# Standard. Achtung: die Express Edition reicht nicht) Microsoft .NET Framework 2.0 Optional*: Microsoft .NET Compact Framework 2.0 SP1, Microsoft .NET Framework SDK 2.0, Acrobat Reader, Microsoft ActiveSync.

*Optionale Software-Produkte, die auf dem Installationsrechner nicht zur Verfügung stehen, können kostenlos über die Internetseiten der entsprechenden Hersteller bezogen werden. Falls diese Produkte nicht installiert werden, stehen einige erweiterte Funktionen von VisiWinNET Professional nicht zur Verfügung.

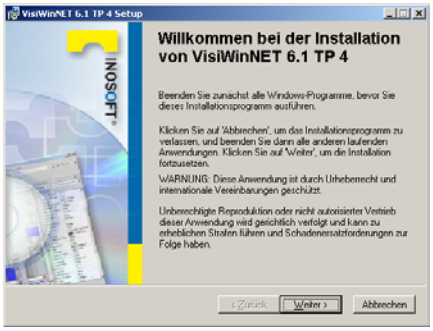
Für die Installation wird weiterer freier Speicherplatz auf der Festplatten benötigt.

2.1.3 Installationsanweisung

Nach Einlegen der VisiWinNET-CD in das entsprechende Laufwerk wird automatisch das Installationsprogramm gestartet.

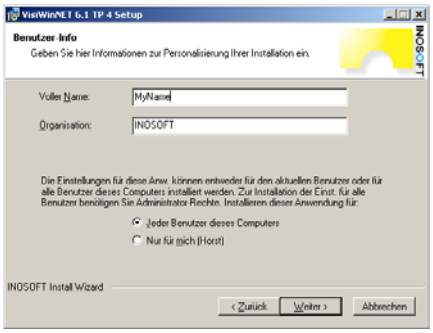
Beachten Sie, dass Visual Studio .NET bereits installiert sein muss! Das Setup-Programm speichert hier den Pfad des Formulardesigner-Tools, das durch den Projektmanager aufgerufen werden soll.

Der Installationsassistent führt den Benutzer durch die einzelnen Schritte der Installation. Dabei wird jeweils über die "Weiter"-Schaltfläche zum nächsten Schritt gewechselt.


VisiWinNET 6.1 TP 4 Setup
Willkommen bei der Installation von VisiWinNET 6.1 TP 4
INOSOFT
Beenden Sie zunächst alle Windows-Programme, bevor Sie dieses Installationsprogramm ausführen.
Klicken Sie auf 'Abbrechen', um das Installationsprogramm zu verlassen, und beenden Sie dann alle anderen laufenden Anwendungen. Klicken Sie auf 'Weiter', um die Installation fortzusetzen.
WARNUNG: Diese Anwendung ist durch Urheberrecht und internationale Vereinbarungen geschützt.
Unberechtigte Reproduktion oder nicht autorisierter Vertrieb dieser Anwendung wird gerichtlich verfolgt und kann zu erheblichen Strafen führen und Schadenersatzforderungen zur Folge haben.
< Zurück
Weiter >
Abbrechen

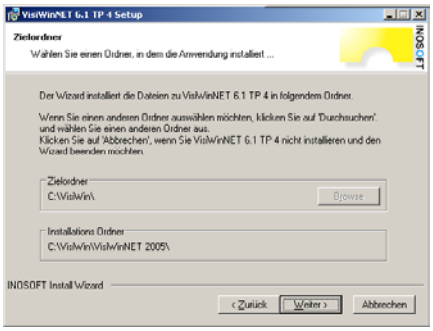
Willkommen

Der Dialog "Willkommen" beschreibt die Funktionsweise des Installationsassistenten und der "Weiter"-Schaltfläche.


VisiWinNET 6.1 TP 4 Setup
Benutzer-Info
Geben Sie hier Informationen zur Personalisierung Ihrer Installation ein.
INOSOFT
Voller Name:
MyName
Organisation:
INOSOFT
Die Einstellungen für diese Anw. können entweder für den aktuellen Benutzer oder für alle Benutzer dieses Computers installiert werden. Zur Installation der Einst. für alle Benutzer benötigen Sie Administrator-Rechte. Installieren dieser Anwendung für:
Jeder Benutzer dieses Computers
Nur für mich (Horst)
INOSOFT Install Wizard
< Zurück
Weiter >
Abbrechen

Benutzerinfo

Fragt allgemeine Benutzerinformationen ab. Die Optionen "Jeder Benutzer dieses Computers" und "nur für mich" legen fest, welche Benutzer das Programm verwenden können.

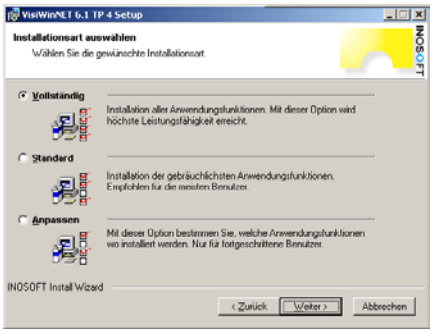

VisiWinNET 6.1 TP 4 Setup
Zielordner
Wählen Sie einen Ordner, in dem die Anwendung installiert ...
Der Wizard installiert die Dateien zu VisiWinNET 6.1 TP 4 in folgendem Ordner.
Wenn Sie einen anderen Ordner auswählen möchten, klicken Sie auf 'Durchsuchen' und wählen Sie einen anderen Ordner aus.
Klicken Sie auf 'Abbrechen', wenn Sie VisiWinNET 6.1 TP 4 nicht installieren und den Wizard beenden möchten.
Zielordner
C:\VisiWin\
Browse
Installations Ordner
C:\VisiWin\VisiWinNET 2005\
INOSOFT Install Wizard
< Zurück
Weiter >
Abbrechen

Zielverzeichnisse wählen

Der Zielordner beschreibt das Verzeichnis, in dem die VisiWin-Produkte installiert werden. Diese Einstellung ist nicht verfügbar, wenn bereits ein VisiWin-Produkt installiert wurde.

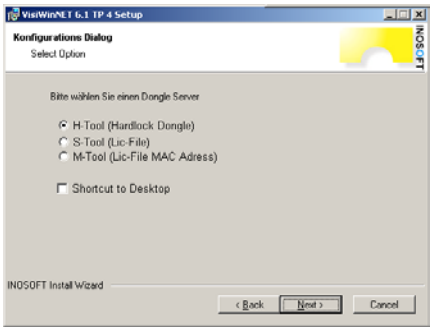
Der Installationspfad beschreibt das Verzeichnis, in dem die aktuelle Installation vorgenommen wird.

Die Schaltflächen "Durchsuchen" öffnen einen Dialog zur komfortablen Auswahl eines Verzeichnisses.


VisiWinNET 6.1 TP 4 Setup
Installationsart auswählen
Wählen Sie die gewünschte Installationsart.
Vollständig
Installation aller Anwendungsfunktionen. Mit dieser Option wird höchste Leistungsfähigkeit erreicht.
Standard
Installation der gebräuchlichsten Anwendungsfunktionen. Empfohlen für die meisten Benutzer.
Anpassen
Mit dieser Option bestimmen Sie, welche Anwendungsfunktionen wo installiert werden. Nur für fortgeschrittene Benutzer.
INOSOFT Install Wizard
< Zurück
Weiter >
Abbrechen

Installationsart auswählen

Hier kann gegebenenfalls die Installation abgespeckt werden.



Konfiguration

Legt die Lizenzierung fest.

2.1.4 Softwareschutz

VisiWinNET unterliegt Lizenzierungsbedingungen. Sowohl die Entwicklungsumgebung als auch die Laufzeit sind stückzahl- und leistungsumfanggebunden kostenpflichtig. Die Software ist gegen Missbrauch und unentgeltliche Verwendung geschützt. Der Kunde kann zwischen verschiedenen Schutzmaßnahmen wählen, die sich in Hinblick auf Hardwareressourcen und Großserienfertigung unterscheiden:

Schutz	Beschreibung
LPT-Dongle	Dongle für die Druckerschnittstelle.
Bus-Dongle	Dongle für einen PCI-Steckplatz.
USB-Dongle	Dongle für die USB-Schnittstelle.
Lic-File	Vertragsabhängige Freischaltung über Lizenzierungsdatei.

Die Dongle unterscheiden sich im Leistungsumfang

Entwicklungsdongle Freischaltung der Entwicklungsumgebung und des Laufzeitsystems mit folgenden Einschränkungen:

- 20000 Prozesswerte können mit den Steuerungen ausgetauscht werden.
- Ein Client-Server-Projekt läuft 4 Stunden. Danach ist ein Neustart nötig.

Laufzeitdongle Der Laufzeit-Softwareschutz ist leistungsumfang- und projekttypenabhängig gestaffelt. Entscheidend dabei ist die Anzahl der zwischen SPS und Visualisierung auszutauschenden Prozesswerte sowie die Client-Serverkonstellation und die verwendete Produktvariante. Die Staffelung ist mit den dazugehörigen Preisen aus der aktuellen Preisliste zu entnehmen.

Toolboxoption Die Toolboxoption wird im Laufzeit-Softwareschutz implementiert. Sie ermöglicht die Änderung der Projektdatenbank auf dem Zielsystem. So können z.B. Änderungen von Meldedefinitionen und mehrsprachigen Texten direkt vom Kunden durchgeführt werden.

Einschränkungen ohne Softwareschutz

VisiWinNET ist ohne Softwareschutz nur eingeschränkt nutzbar. Sowohl die Entwicklungsumgebung als auch die Laufzeit befinden sich dann im "Demo-Modus". Dieser Modus erlaubt den Start und die begrenzte Bearbeitung der mitgelieferten Demo-Projekte.

Der Demo-Modus enthält folgende Einschränkungen:

- Ohne Dongle kann in der Entwicklungsumgebung nur ein Projekt mit dem Namen "VWNTest" angelegt und bearbeitet werden.
- Der Projektexplorer in der Entwicklungsumgebung zeigt in der Statuszeile den Demomodus an.
- Bei jedem Start der Entwicklungsumgebung wird auf den fehlenden Softwareschutz hingewiesen.
- Das Laufzeitsystem schaltet sich nach einer Stunde ab.

2.2 Schnelleinstieg

Das folgende Kapitel vermittelt als praktisches Beispiel einen Überblick über die Funktionsweise der Entwicklungsumgebung, des Laufzeitsystems und der einzelnen Systeme, die spezielle visualisierungstypische Aufgaben übernehmen.

Das folgende Testszenario ermöglicht den Aufbau einer voll funktionsfähigen Applikation. Die Gliederung des Szenarios entspricht der Einfachheit halber teilweise der logischen Aufteilung von VisiWinNET, teilweise auch praktischen Grundsätzen der Visualisierungsentwicklung:

Steuerungs-anbindung mit OPC-Servern VisiWin-Treibern oder VisiWinNET-Treibern

Herstellung der Kommunikation mit der SPS: VisiWinNET unterstützt drei verschiedene Arten von Kommunikationskomponenten:

- OPC-Server sind als standardisierte Kommunikationskomponenten für mittlerweile fast jede Steuerung erhältlich und binden den Projekteur nicht an ein spezifisches Visualisierungsprodukt, da mittlerweile fast jedes Visualisierungspaket den OPC-Standard beherrscht.
- VisiWin-Treiber als Ergebnis langjähriger Entwicklung insbesondere auch für langjährige Kunden oder Umsteigern von älteren Produkten (VisiWin32/VisiWinStudio). Sie stehen als Alternative zu OPC-Servern weiterhin zur Verfügung, gegebenenfalls auch für Steuerungen, für die kein OPC-Server existiert und für die im Rahmen einer Sonderentwicklung ein Treiber erstellt wurde.
- VisiWinNET-Treiber sind die Nachfolger der VisiWin-Treiber. Sie wurden auf Basis des .NET Framework implementiert und garantieren damit eine größtmögliche Durchgängigkeit in den unterschiedlichen Betriebssystemen. VisiWinNET-Treiber beinhalten optional zur Datenübernahme einen "Browser"-ähnlichen Mechanismus und weitere Unterstützungen zur Entwicklungszeit. Das Verhalten zur Laufzeit ist speziell auf die Beseitigung von Synchronisationsproblemen bei OPC-Servern optimiert.

Das Beispiel zeigt alle drei Anbindungsarten anhand der beim Produkt beiliegenden Demonstrationskomponenten

Die Laufzeitsysteme "Compact" und "Standard" unterstützen jeweils nur eine Teilmenge von VisiWin-Treibern:

- Die "klassischen" VisiWin-Treiber werden von der "Compact"-Laufzeit nicht unterstützt
- Die neuere Generation von VisiWinNET-Treibern kann auch unter Compact verwendet werden.

Darstellung und Manipulation von Prozesswerten in der Applikation

Die Applikation wird mit Leben gefüllt. Erste, sich ändernde Prozesswerte werden auf unterschiedliche Arten dargestellt. Das Schreiben von Prozesswerten wird mithilfe spezieller Steuerelemente vorgenommen.

Grundregeln

Hier wird gezeigt, wie eine Applikation modular aufgebaut

Applikationsaufbau	wird. Die Unterteilung und der Wechsel zwischen den einzelnen Bildschirmansichten wird erklärt.
Sprachumschaltung	Die Schnittstelle zur Sprachumschaltung wird mit Daten ausgefüllt und in Betrieb genommen.
Alarmsystem	Fehlerzustände ermitteln und darstellen.
Archivsystem	Sinnvolle Langzeitmessungen an Prozesswerten.
Rezepte	Parametersätze darstellen, optimieren und zur Maschine senden.
Protokollierung	Alles muss nachvollziehbar sein. Aufzeichnen von Benutzerinteraktionen und SPS-Werten, so dass eine Chronik der Maschine entsteht.
Benutzerverwaltung	Verschiedene Benutzer dürfen nur begrenzten Zugriff auf die Bestandteile der Applikation erlangen.

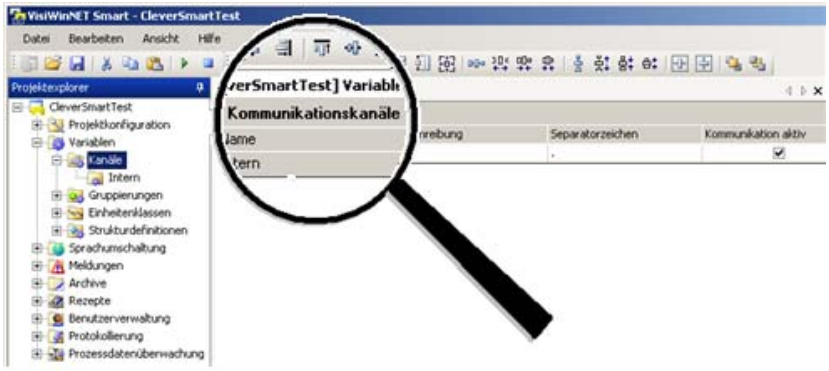
2.2.1 Steuerungsanbindung mit OPC

In diesem Kapitel wird das Anlegen eines neuen OPC-Servers und die Übernahme von OPC-Items in das VisiWinNET-Testprojekt beschrieben.

Ein wesentlicher Bestandteil einer Visualisierung unter VisiWinNET ist die Projektierung der Prozessanbindung. Hier wird festgelegt, welche Daten aus der SPS für die Visualisierung benötigt werden, mit welchen Kommunikationseinstellungen sie ausgetauscht werden und, wenn Bedarf besteht, wie diese Daten in der Visualisierung benannt werden. Die Prozessanbindung wird im Variableneditor vorgenommen. Mit dem Variableneditor werden Definitionen festgelegt, die bestimmen, welche Prozesswerte mit der SPS ausgetauscht werden sollen. Der Variablenkernel (zentrale Komponente zum Austausch von Prozesswerten zwischen SPS und Visualisierung) interpretiert diese Definitionen zur Laufzeit und stellt den entsprechenden Kontakt mit den Kommunikationskomponenten her. Der Variableneditor wird durch den Knoten "Variablen" repräsentiert.

Der Knoten "Variablen" im VisiWinNET Projektexplorer ist zu expandieren. Über einen Klick auf den darunter liegenden Knoten "Kanäle" ist der Tabelleneditor des Variableneditors zu aktivieren.

Ist im Projektexplorer der Knoten "Kanäle" markiert, werden im Tabelleneditor alle Kommunikationskanäle des Projektes aufgelistet.



Auf dem Tabelleneditor ist das Kontextmenü zu öffnen (Klick mit rechter Maustaste) und hier der Eintrag "Neu" zu wählen.

Diese Aktion öffnet den Dialog "Kanal hinzufügen". Hier werden grundlegende Eigenschaften des neuen Kanals festgelegt:

- Name: Ein wahlfreier Name für den Kanal. Der Name muss innerhalb der Kommunikationskanäle des Projektes eindeutig sein. Verweise auf Prozessvariablen in der Applikation beziehen diesen Namen ein (z.B. "Ch1.w0" bezeichnet die Variable "w^0" im Kommunikationskanal mit dem Namen "Ch1".
- Kanaltyp: Entweder OPC-Server oder Treiber. Diese Einstellung ist auf "OPC-Server" zu stellen.

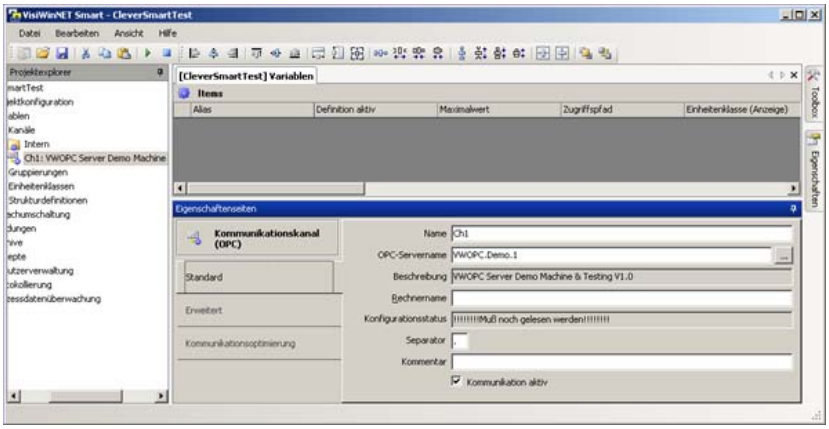
OPC-Server/ Treiber: Über die -Schaltfläche ist der Dialog zur Auswahl einer Kommunikationskomponente zu öffnen. Hier ist unter dem Knoten "Arbeitsplatz→OPC-Server" der Eintrag "VWOPC Server Demo Machine & Testing" auszuwählen.

Nach dem Schließen des Dialoges ist auf den Knoten "Kanäle" im Projektexplorer zu klicken.

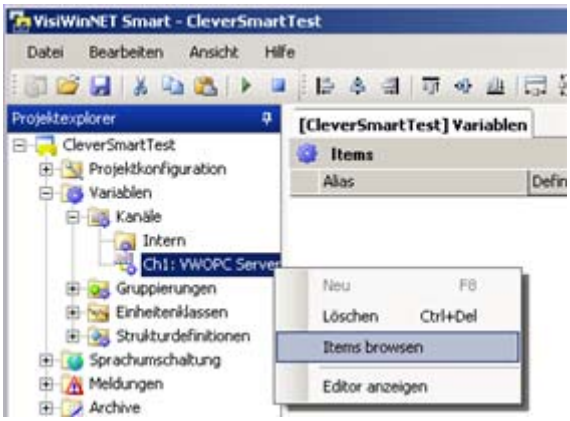
Die Markierung des "Kanäle"-Knotens bewirkt das Speichern der eingegebenen Daten in die Projektdatenbank.

Der neue Kommunikationskanal wird nun durch einen Knoten im Projektexplorer repräsentiert.

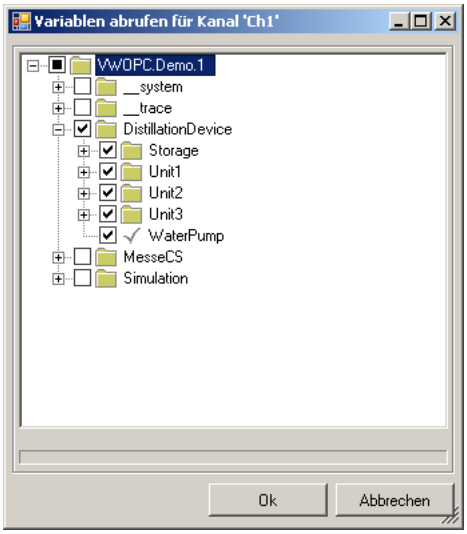
Wird der Knoten markiert, blendet sich automatisch die VisiWinNET-Eigenschaftenseite ein. Hier wird der vollständige Parametersatz des Kommunikationskanals zum Bearbeiten angeboten.



Über das Kontextmenü des Kommunikationskanals ist die "Browsen"-Funktion zu aktivieren.



Es öffnet sich nun ein Fenster mit dem Titel "Variablen abrufen für Kanal 'Ch1'".



Der Variablenbrowser zeigt alle im OPC-Server befindlichen Itemdefinitionen hierarchisch an. In der obersten Hierarchie befindet sich der Name des OPC-Servers, in der untersten Hierarchie werden die einzelnen Items dargestellt. Dazwischen befinden sich die sogenannten "Branches". Diese stellen kein Objekt dar, sondern zeigen nur die Strukturierung der Items in Namensräumen an:

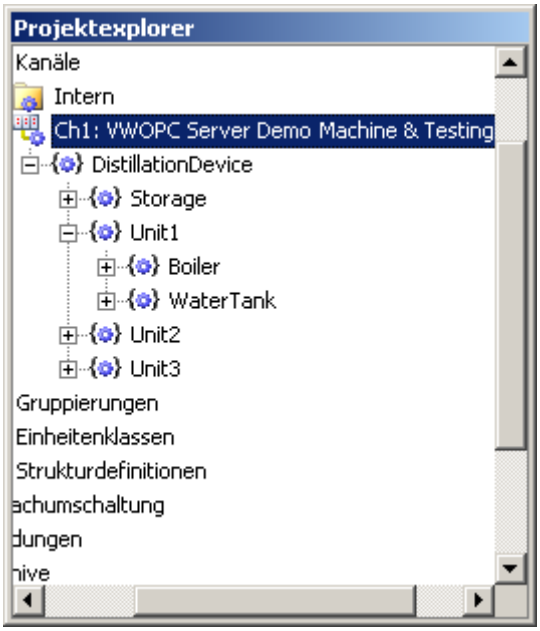
Itembezeichner werden durch ein Trennzeichen (ähnlich wie in der vollständigen Pfadangabe einer Datei das "\"-Zeichen) in Bezeichnerteile zerlegt. Items, die mit gleichen Bezeichnerteilen beginnen, werden unter einem gemeinsamen Branch zusammengefasst.

Wird ein Namensraum durch das Kontrollkästchen aktiviert, so werden dadurch auch alle darunter liegenden Namensräume und Itembezeichner mit aktiviert.

Durch die Auswahl eines Itembezeichners wird die entsprechende Variablendefinition in die VisiWinNET-Projektdatenbank übernommen.

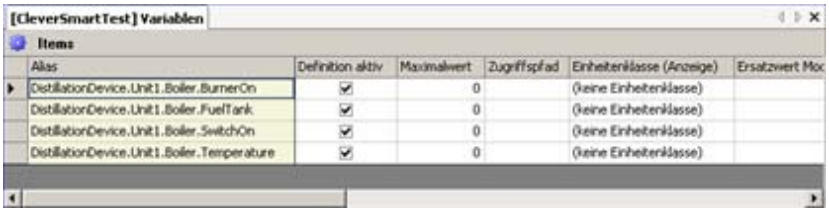
Der Knoten "Distillation Device" ist, wie in der Grafik dargestellt, auszuwählen. Die Auswahl ist über die "OK"-Schaltfläche zu bestätigen.

Die gewählten Items werden als Variablendefinitionen in die Projektdatenbank übernommen. Die Darstellung im Projektexplorer beschränkt sich auf die hierarchische Ansicht von Namensräumen.



Die Variablendefinitionen werden mitsamt ihrer Parameter im Tabelleneditor angezeigt.

Ein Klick auf einen Namensraum öffnet den Tabelleneditor in der Entwicklungsumgebung (über dem Designbereich).



Hier werden die Variablendefinitionen des gerade gewählten Namensraumes tabellarisch aufgelistet. Für das hier beschriebene Beispiel reicht es jedoch, die Variablendefinitionen aus dem OPC-Server zu übernehmen. Weitere Anpassungen sind nicht nötig. Der Tabelleneditor kann also wieder geschlossen werden.

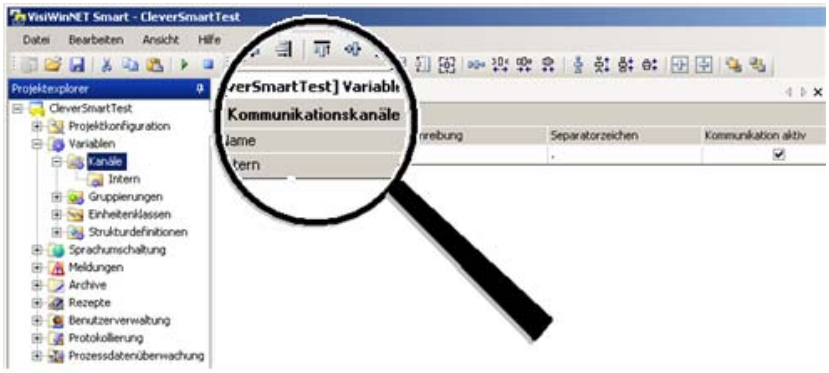
2.2.2 Steuerungsanbindung mit VisiWin-Treibern

In diesem Kapitel wird das Anlegen eines neuen VisiWin-Treibers und den entsprechenden Variablendefinitionen beschrieben. Außerdem wird die Funktion von Strukturen erläutert.

VisiWin-Treiber sind eine Alternative zu OPC-Servern. Sie werden üblicherweise verwendet, wenn kein OPC-Server für eine Steuerung vorliegt, wenn sehr spezielle Anforderungen über einen Kundenauftrag realisiert werden oder wenn bestehende Anbindungen aus älteren VisiWin-Visualisierungspaketen ohne Änderung übernommen werden sollen.

Der Knoten "Variablen" im VisiWinNET Projektexplorer ist zu expandieren. Über einen Klick auf den darunter liegenden Knoten "Kanäle" ist der Tabelleneditor des Variableneditors zu öffnen.

Ist im Projektexplorer der Knoten "Kanäle" markiert, werden im Tabelleneditor alle Kommunikationskanäle des Projektes aufgelistet.



Auf dem Tabelleneditor ist das Kontextmenü zu öffnen (Klick mit rechter Maustaste) und hier der Eintrag "Neu" zu wählen.

Diese Aktion öffnet den Dialog "Kanal hinzufügen". Hier werden grundlegende Eigenschaften des neuen Kanals festgelegt:

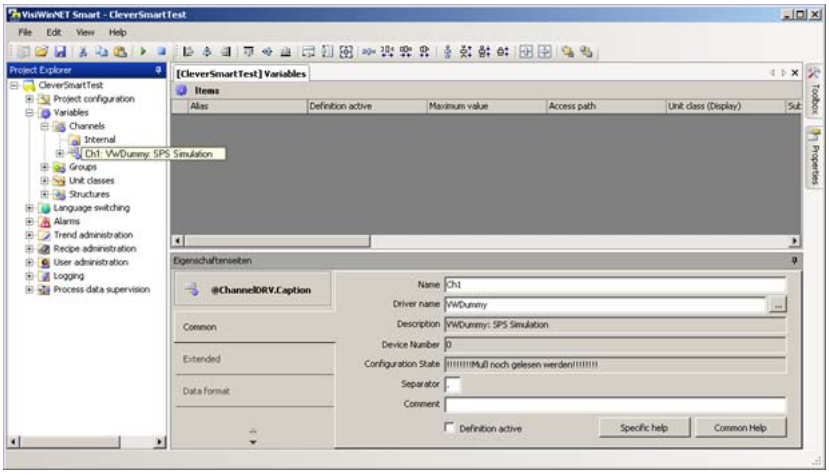
- Name: Ein wahlfreier Name für den Kanal. Der Name muss innerhalb der Kommunikationskanäle des Projektes eindeutig sein. Verweise auf Prozessvariablen in der Applikation beziehen diesen Namen ein (z.B. "Ch1.w0" bezeichnet die Variable "wo" im Kommunikationskanal mit dem Namen "Ch1".
- Kanaltyp: Entweder OPC-Server oder Treiber. Diese Einstellung ist auf "VisiWin-Treiber" zu stellen.
- OPC-Server/ Treiber: Über die ...-Schaltfläche ist der Dialog zur Auswahl einer Kommunikationskomponente zu öffnen. Hier ist der Eintrag "VWDummy SPS-Simulation" auszuwählen.

Nach dem Schließen des Dialoges ist auf den Knoten "Kanäle" im Projektexplorer zu klicken.

Die Markierung des "Kanäle"-Knotens bewirkt das Speichern der eingegebenen Daten in die Projektdatenbank.

Der neue Kommunikationskanal wird nun durch einen Knoten im Projektexplorer repräsentiert.

Wird der Knoten markiert, blendet sich automatisch die VisiWinNET-Eigenschaftenseite ein. Hier wird der vollständige Parametersatz des Kommunikationskanals zum Bearbeiten angeboten.



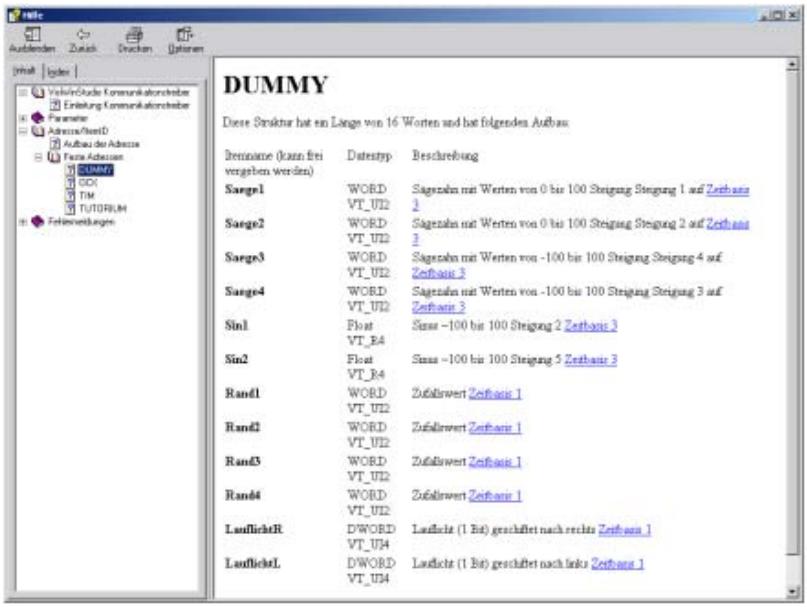
Damit ist die Wahl der Kommunikationskomponente getroffen. Über die Definition des Kommunikationskanals wird der Kontakt zur Kommunikationskomponente hergestellt. Im Folgenden sind noch die Festlegungen der auszutauschenden Prozesswerte vorzunehmen. Dies geschieht über die Definition von Variablen.

Variablendefinitionen in Treibern sind nicht automatisiert anzulegen. Sie müssen per Hand durch das Hinzufügen einzelner Datensätze im Tabelleneditor definiert werden.

Ein spezifisches Merkmal sind die Adressen, über die der Treiber Daten mit der SPS austauscht. Jeder Variablen muss eine Adressangabe und (über den Datentyp) eine Länge zugewiesen werden. Dazu kommt, dass Adressangaben je nach SPS-Typ symbolisch oder direkt zu verwenden sind. Verschiedene Treiber erwarten also unterschiedliche Adressangaben. Die Information, welcher Syntax die Angabe einer SPS-Adresse folgt, ist in der spezifischen Treiberhilfe dokumentiert. Der schnellste Weg zu dieser Hilfe geht über die Schaltfläche "spezifische Hilfe" in den Kanaleigenschaften.

Über die Schaltfläche "spezifische Hilfe" auf dem Karteireiter "Standard" ist die Hilfe zum Treiber zu öffnen.

Alle Treiberhilfen enthalten einen Abschnitt über den Aufbau der spezifischen Adressangaben (Adresse/ItemID)



Was bei einer realen SPS erst durch die Programmierung festgelegt wird, ist im Dummy-Treibers bereits fest verdrahtet. Über die hier angegeben Adressen können simulierte Variablenwerte abgerufen werden. Der Abschnitt "Adresse/ItemID→feste Adressen→Dummy" beschreibt die Datenstruktur der "Dummy"-Adresse.

Wieso "Datenstruktur der Dummy"-Adresse? Sind "Saege1", "Saege2" usw. denn keine Adressen?

Nein, sind sie nicht. Treiber haben grundsätzlich die Möglichkeit, mit der Steuerung Daten einzeln oder blockweise auszutauschen. Die Definition einer Variablen als Grunddatentyp "VT_..." ermöglicht der Zugriff auf eine einzelne Prozessvariable. Die Definition einer Variablen vom Datentyp einer Struktur ist ein blockweiser Datenaustausch. Innerhalb des Blockes liegen die einzelnen Variablen, deren Länge durch den Datentyp bestimmt wird. Eine Variable vom Datentyp VT_I4 z.B. hat eine feste Länge von vier zu übertragende Bytes. Die Übertragung eines Blockes erfordert nur eine steuerungsspezifische Startadresse und die Länge des Datenblockes, die sich im Fall von Strukturen über die Länge der einzelnen Elemente berechnet.

Eine Struktur bildet sich aus einzelnen Elementen, die wiederum Strukturen oder Grunddatentypen sein können. Strukturen haben den Vorteil, dass mit einem minimalen Protokoll-Overhead viele Daten gelesen oder geschrieben werden können. Dies ist insbesondere dann zwingend erforderlich, wenn die SPS-Schnittstelle langsam ist (z.B. seriell 9600 Baud). Über Strukturen lässt sich also die Kommunikation optimieren. Für die Übertragung einzelner SPS-Variablen muss nicht jedes mal ein Protokollrahmen

gesendet werden. Über einen einzigen Protokollrahmen kann bei entsprechender Projektierung beispielsweise ein ganzer Datenbaustein transferiert werden. Die Auflösung in die einzelnen Elemente oder Variablen übernimmt das Laufzeitsystem in der Applikation.

Strukturen werden in VisiWinNET auf folgendem Wege definiert.

Über das Kontextmenü des Knotens "Strukturdefinitionen" ist eine neue Struktur zu definieren.

Der Name der neuen Struktur ist in der VisiWinNET-Eigenschaftenseite in "sDummy" zu ändern.

Im Tabelleneditor der neuen Definition sind 12 neue Strukturelemente anzulegen (Klick auf die leere Tabelle anschließend 12 x "F8" drücken.

Die Parameter der neuen Strukturelemente sind wie in der Hilfe angegeben zu ändern:

Name	Datentyp
Saege1	**VT_UI2**
Saege2	**VT_UI2**
Saege3	**VT_UI2**
Saege4	**VT_UI2**
Sin1	**VT_R4**
Sin2	**VT_R4**
Rand1	**VT_UI2**
Rand2	**VT_UI2**
Rand3	**VT_UI2**
Rand4	**VT_UI2**
LauflichtR	**VT_UI4**
LauflichtL	**VT_UI4**

Abschließend ist zu überprüfen, ob der Parameter "Ordnungsnummer" fortlaufend in oben genannter Reihenfolge aufsteigt.

Die hier aufgebaute Struktur beschreibt den Aufbau eines Datenblockes. Die Struktur selbst ist aber noch kein Baustein, der selbst Daten austauscht. Er ist nur eine Typendeklaration. Diese Deklaration kann in den Variablendefinitionen als Datentyp verwendet werden.

Im Projektexplorer ist zum Knoten "Ch1: VWDummy..." zu wechseln.

Die Ansicht im Tabelleneditor zeigt die Variablendefinitionen des Kanals. Da noch keine Variablen definiert wurden, ist die Tabelle leer.

Über das Kontextmenü des Tabelleneditors ist eine neue Prozessvariable anzulegen. In dieser neuen Variablen sind folgende Werte zu ändern:

Parameter	Wert
Name	DummyItem
Adresse	Dummy
Datentyp	sDummy

Der Name der Variablen ist grundsätzlich wahlfrei. Ein zweckgebundener Name, aus dem die Funktion einer Variablen hervorgeht, erleichtert in den folgenden Entwicklungsschritten die Projektierung enorm.

Die angegebene Adresse bezieht sich auf die in der Hilfe beschriebene Funktion des Treibers.

Als Datentyp wird in der Auswahlliste die definierte Struktur angeboten. Die Definition einer Variablen vom Datentyp einer Struktur bewirkt zur Laufzeit folgenden Ablauf:

Während der Initialisierungsphase lädt der Variablenkern die Variablendefinitionen aus der Projektdatenbank und baut ein entsprechendes Speicherabbild auf. Dazu ermittelt er den Aufbau und die Länge der als Datentypen angegebenen Strukturen.

Bei der Anforderung von Werten aus der Applikation übersendet der Variablenkern die Startadresse und die Länge der Variablen. Im oben genannten Beispiel ergibt sich die Länge des angeforderten Datenblockes mit 32 Byte.

Der Treiber liest die angegebene Anzahl von Bytes ab der übergebenen Startadresse mit einem Protokollzugriff und gibt den Byteblock an den Variablenkern zurück.

Der Variablenkern ordnet jetzt die einzelnen Bytes intern zu den entsprechenden Variablen im Speicherabbild zu.

2.2.3 Steuerungsanbindung mit VisiWinNET-Treibern

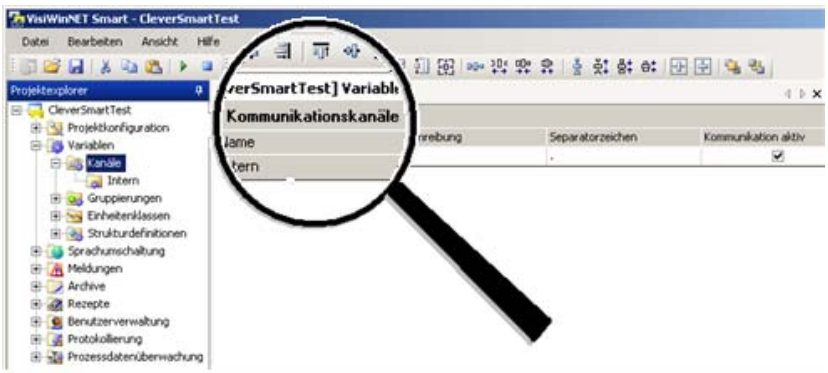
In diesem Kapitel wird das Anlegen eines neuen VisiWinNET-Treiber und die Übernahme von Variablendefinitionen in das VisiWinNET-Testprojekt beschrieben.

Ein wesentlicher Bestandteil einer Visualisierung unter VisiWinNET ist die Projektierung der Prozessanbindung. Hier wird festgelegt, welche Daten aus der SPS für die Visualisierung benötigt werden, mit welchen Kommunikationseinstellungen sie ausgetauscht werden und, wenn Bedarf besteht, wie diese Daten in der Visualisierung benannt werden. Die Prozessanbindung wird im Variableneditor vorgenommen. Mit dem Variableneditor werden Definitionen festgelegt, die bestimmen, welche Prozesswerte mit der SPS ausgetauscht werden sollen. Der Variablenkernel (zentrale Komponente zum Austausch von Prozesswerten zwischen SPS und Visualisierung) interpretiert diese Definitionen zur Laufzeit und stellt den entsprechenden Kontakt mit den Kommunikationskomponenten her. Der Variableneditor wird durch den Knoten "Variablen" repräsentiert.

Viele VisiWinNET-Treiber unterstützen eine Browser-ähnliche Funktion. Damit können die Variablen automatisiert aus dem SPS-Programm in die Visualisierung übernommen werden. Unterstützt der Treiber das Browsen nicht, müssen die Definitionen laut spezifischer Dokumentation des Treibers manuell im Projekt eingegeben werden (Siehe Kapitel "Steuerungsanbindung mit VisiWin-Treibern").

Der Knoten "Variablen" im VisiWinNET Projektexplorer ist zu expandieren. Über einen Klick auf den darunter liegenden Knoten "Kanäle" ist der Tabelleneditor des Variableneditors zu aktivieren.

Ist im Projektexplorer der Knoten "Kanäle" markiert, werden im Tabelleneditor alle Kommunikationskanäle des Projektes aufgelistet.



Auf dem Tabelleneditor ist das Kontextmenü zu öffnen (Klick mit rechter Maustaste) und hier der Eintrag "Neu" zu wählen.

Diese Aktion öffnet den Dialog "Kanal hinzufügen". Hier werden grundlegende Eigenschaften des neuen Kanals festgelegt:

- Name: Ein wahlfreier Name für den Kanal. Der Name muss innerhalb der Kommunikationskanäle des Projektes eindeutig sein. Verweise auf Prozessvariablen in der Applikation beziehen diesen Namen ein (z.B. "Ch1.w0" bezeichnet die Variable "w0" im Kommunikationskanal mit dem Namen "Ch1".
- Kanaltyp: Entweder OPC-Server oder Treiber. Diese Einstellung ist auf "VisiWinNET-Treiber" zu stellen.

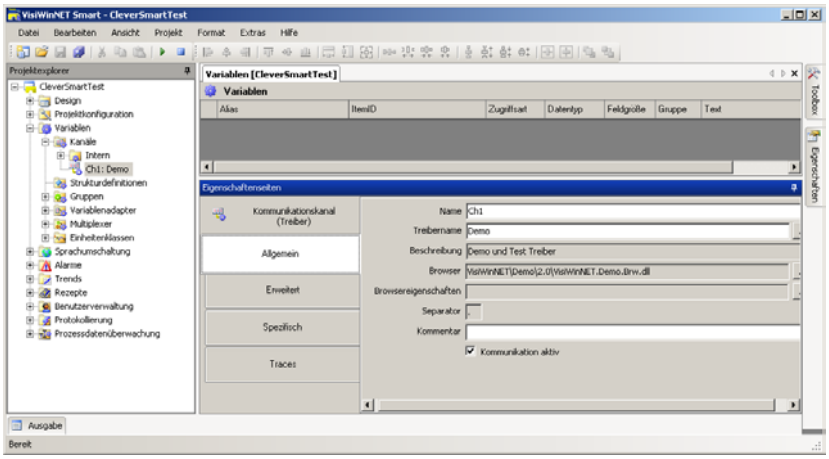
OPC-Server/ Treiber: Über die -Schaltfläche ist der Dialog zur Auswahl einer Kommunikationskomponente zu öffnen. Hier ist aus der Liste der installierten Treiber der Eintrag "Demo, Demo & Test Treiber" auszuwählen.

Nach dem Schließen des Dialoges ist auf den Knoten "Kanäle" im Projektexplorer zu klicken.

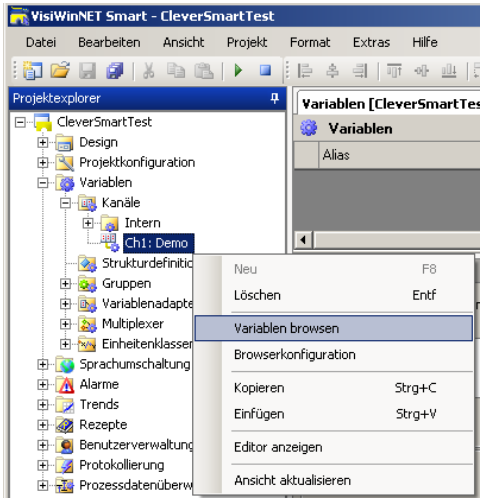
Die Markierung des "Kanäle"-Knotens bewirkt das Speichern der eingegebenen Daten in die Projektdatenbank.

Der neue Kommunikationskanal wird nun durch einen Knoten im Projektexplorer repräsentiert.

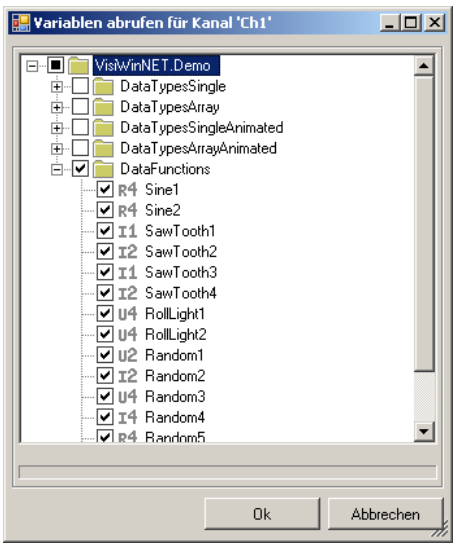
Wird der Knoten markiert, blendet sich automatisch die VisiWinNET-Eigenschaftenseite ein. Hier wird der vollständige Parametersatz des Kommunikationskanals zum Bearbeiten angeboten.



Über das Kontextmenü des Kommunikationskanals ist die "Browsen"-Funktion zu aktivieren.



Es öffnet sich nun ein Fenster mit dem Titel "Variablen abrufen für Kanal 'Ch1'".



Der Variablenbrowser zeigt alle im VisiWinNET-Treiber befindlichen Variablen hierarchisch an. In der obersten Hierarchie befindet sich der Name des Treibers, in der untersten Hierarchie werden die einzelnen Prozessvariablen dargestellt. Dazwischen befinden sich die sogenannten "Branches". Diese stellen kein Objekt dar, sondern zeigen nur die Strukturierung der Prozessvariablen in Namensräumen an:

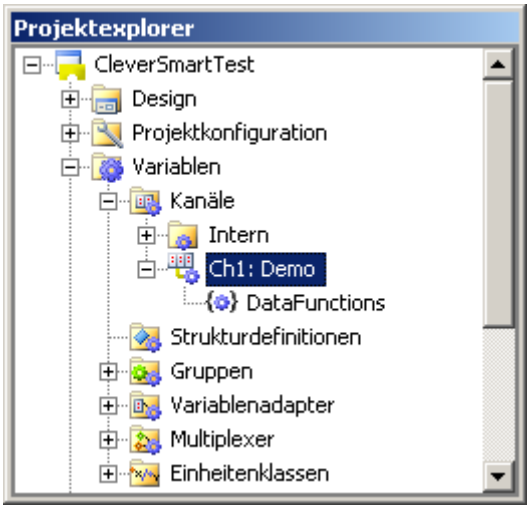
Variablennamen werden durch ein Trennzeichen (ähnlich wie in der vollständigen Pfadangabe einer Datei das "\"-Zeichen) in Bezeichnerteile zerlegt. Prozessvariablen, die mit gleichen Bezeichnerteilen beginnen, werden unter einem gemeinsamen Branch zusammengefasst.

Wird ein Namensraum durch das Kontrollkästchen aktiviert, so werden dadurch auch alle darunter liegenden Namensräume und Variablennamen mit aktiviert.

Durch die Auswahl eines Variablennamens wird die entsprechende Prozessvariablendefinition in die VisiWinNET-Projektdatenbank übernommen.

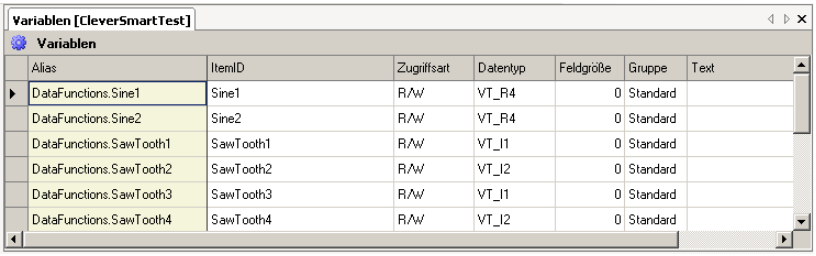
Der Knoten "DataFunctions" ist, wie in der Grafik dargestellt, auszuwählen. Die Auswahl ist über die "OK"-Schaltfläche zu bestätigen.

Die gewählten Variablen werden als Prozessvariablendefinitionen in die Projektdatenbank übernommen. Die Darstellung im Projektexplorer beschränkt sich auf die hierarchische Ansicht von Namensräumen.



Die Prozessvariablendefinitionen werden mitsamt ihrer Parameter im Tabelleneditor angezeigt.

Ein Klick auf einen Namensraum öffnet den Tabelleneditor in der Entwicklungsumgebung (über dem Designbereich).



Alias	ItemID	Zugriffsart	Datentyp	Feldgröße	Gruppe	Text
DataFunctions.Sine1	Sine1	R/W	VT_R4	0	Standard	
DataFunctions.Sine2	Sine2	R/W	VT_R4	0	Standard	
DataFunctions.SawTooth1	SawTooth1	R/W	VT_I1	0	Standard	
DataFunctions.SawTooth2	SawTooth2	R/W	VT_I2	0	Standard	
DataFunctions.SawTooth3	SawTooth3	R/W	VT_I1	0	Standard	
DataFunctions.SawTooth4	SawTooth4	R/W	VT_I2	0	Standard	

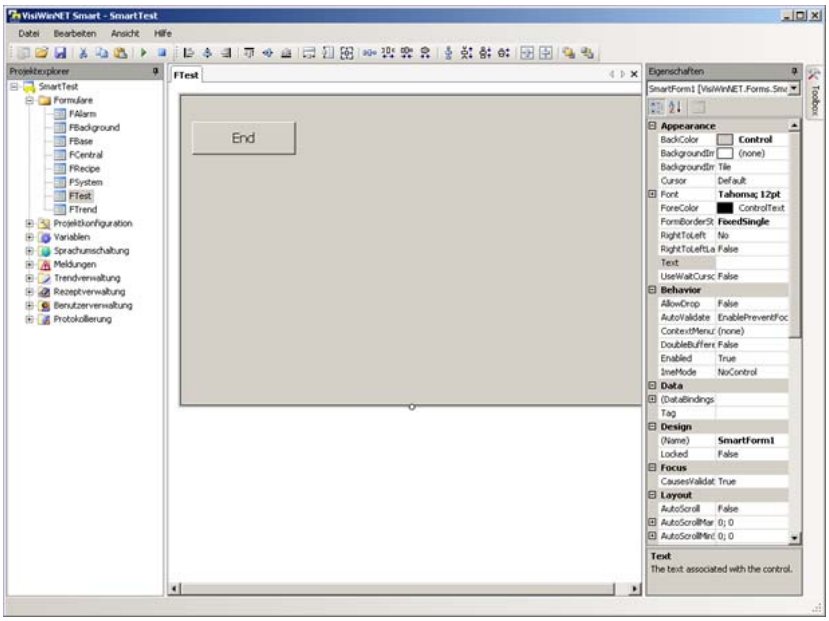
Hier werden die Prozessvariablendefinitionen des gerade gewählten Namensraumes tabellarisch aufgelistet. Für das hier beschriebene Beispiel reicht es jedoch, die Prozessvariablendefinitionen aus dem Treiber zu übernehmen. Weitere Anpassungen sind nicht nötig. Der Tabelleneditor kann also wieder geschlossen werden.

2.2.4 Darstellung und Manipulation von Prozesswerten

Dieses Kapitel beschreibt den allgemeinen Umgang mit Steuerelementen und stellt den Zusammenhang zu den Variablendefinitionen aus der Projektdatenbank dar.

Nach dem Anlegen eines neuen Projektes ist mindestens ein Formular bereits angelegt. Ein Formular repräsentiert eine Bildschirmansicht oder einen Dialog.

Um die Designansicht des Formulars zu öffnen, ist der Knoten "Formulare" zu expandieren. Darunter werden die Formulare des Projektes aufgelistet. Ein Doppelklick auf einen Formularnamen öffnet den Designer.

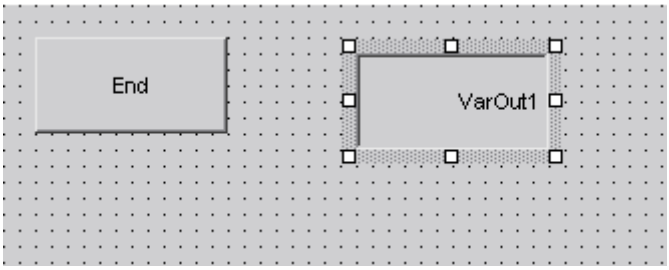


Der Designer stellt eine Bildschirmansicht dar. Das Aussehen der Bildschirmansicht wird über das Platzieren und Parametrieren von Steuerelementen bestimmt. Steuerelemente werden in der Toolbox ausgewählt. In der Grundeinstellung wird das Toolboxfenster oben rechts als Karteireiter dargestellt.

Die Toolbox ist über den entsprechenden Karteireiter zu expandieren. Der Steuerelementetyp "VarOut" ist in der Toolbox zu markieren.

Anschließend ist auf dem Formular die Stelle durch einen Klick mit der linken Maustaste zu wählen, wo das Steuerelement zu platzieren ist. Wird die Maustaste sofort wieder losgelassen, wird das Steuerelement an dieser Stelle mit einer vorgegebenen Größe platziert. Wird hingegen bei gehaltener Maustaste der Mauszeiger bewegt, bildet die Bewegung auf dem Bildschirm einen Rahmen. Das Loslassen der Maustaste bewirkt, dass das Steuerelement mit Größe und Position des dargestellten Rahmens auf dem Formular erzeugt wird.

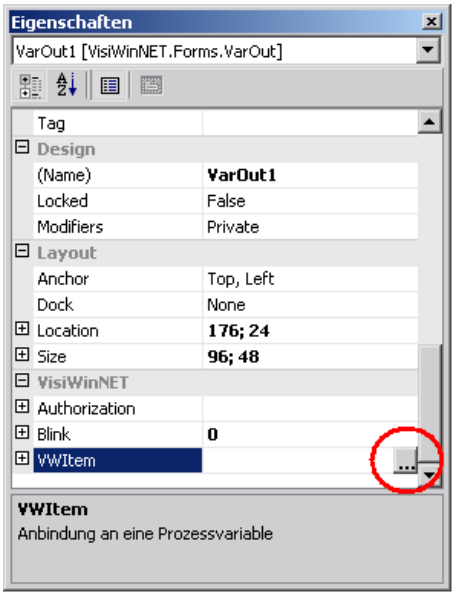
Nach dem Platzieren erscheint das 'VarOut'-Steuerelement auf dem Formular



Der gewählte Steuerelementetyp dient zur numerischen oder alphanumerischen Anzeige von Variablenwerten. Die wesentliche Festlegung, welcher Prozesswert anzuzeigen ist, wird über die Eigenschaft "VWItem" vorgenommen.

Das Steuerelement ist mit einem Mausklick zu markieren, so dass wie oben gezeigt, der Selektionsrahmen sichtbar wird.

Im Eigenschaftenfenster ist über die Bildlaufleiste so weit zu blättern, bis die Eigenschaft "VWItem" sichtbar wird. Das leere Wertefeld der Eigenschaft ist über einen Mausklick zu markieren.

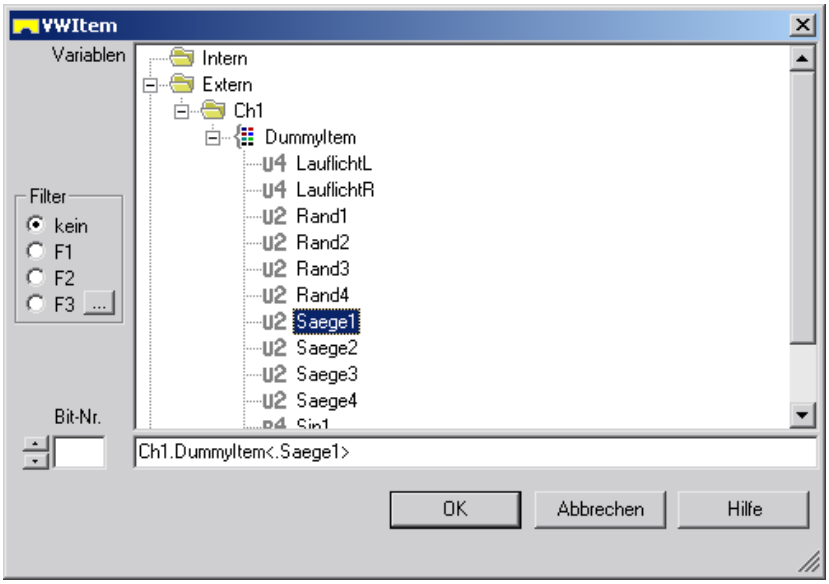


Im Wertefeld der Eigenschaft erscheint eine kleine Schaltfläche, über die ein Dialog zur Auswahl einer Variablen geöffnet werden kann.

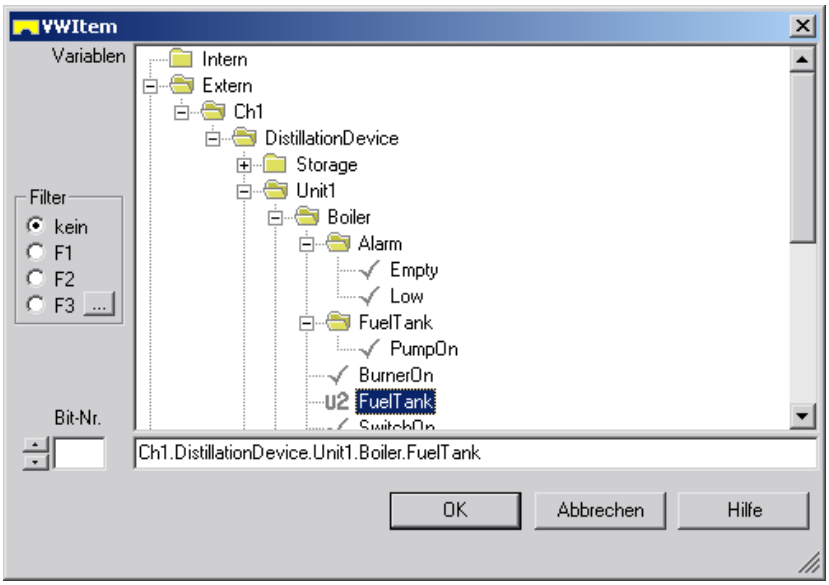
Die oben gezeigte Auswahlschaltfläche ist durch einen Mausklick zu bedienen.

Die vorhergehenden Kapitel beschreiben die Anbindung mit einem OPC-Server und einem VisiWin-Treiber. Je nachdem, welche Anbindung gewählt wurde, zeigt der Dialog folgenden Inhalt

Für VisiWin-Treiber...



...oder für OPC-Server



Der Dialog zeigt die zur Verfügung stehenden Prozessvariablen, die durch Browsen oder selbst Definieren in der Projektdatenbank gespeichert wurden. Im wesentlichen folgt der Aufbau der hier dargestellten Hierarchie dem Aufbau im Projektexplorer.

- "Intern" kennzeichnet Variablen, die keinem Kommunikationskanal zugeordnet sind, die also kein reales Abbild in einer Steuerung haben. Diese Variablen können trotzdem innerhalb einer Visualisierung benötigt werden, um z.B. variablengesteuerte Vorgänge auszulösen oder einfach als Puffer für Werte aus der Visualisierung.

"Extern" beinhaltet die Variablen, die ein SPS-Abbild haben. Unter extern werden die Kommunikationskanäle aufgelistet. Darunter kommen die Namensräume, die auch schon im Browsen verwendet wurden. Es folgen die Variablendefinitionen. Gegebenenfalls werden bei Treibern darunter noch die Strukturelemente zur Verfügung gestellt.

Die in den oben dargestellten Bildern selektierten Variablen sind auszuwählen. Über die OK-Taste ist die Auswahl zu bestätigen.

Durch die Festlegung der 'VWItem'-Eigenschaft ändert sich die Darstellung im Steuerelement "VarOut1". Bei der Festlegung der darzustellenden Variablen wird der Datentyp interpretiert. Die Anzeige in den Steuerelementen springt auf einen Wert, der die Darstellung zur Laufzeit simuliert. Damit ist es möglich, schon zur Entwicklungszeit zu überprüfen, ob das Steuerelement den Wert überhaupt vollständig anzeigen kann oder ob ggf. die Steuerelementeabmessungen geändert werden müssen.

Ein abschließender Test zeigt die Funktionsfähigkeit der Anbindung.

Der Testlauf eines Standard-Projektes gestaltet sich einfach:

In der Werkzeugleiste ist die -Schaltfläche zu drücken.

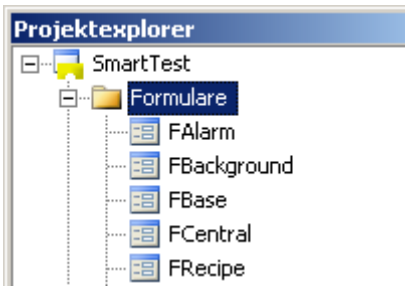
Das Projekt startet. Die nun angezeigte Ansicht entspricht der Oberfläche, die dem Visualisierungsbenutzer angezeigt wird. Ist die Funktion erfolgreich, sollte in der Darstellung nun ein sich ändernder Wert angezeigt werden.

VisiWinNET stellt eine Reihe weiterer Steuerelemente zur Verfügung, die die Anzeige oder Eingabe grafisch oder alphanumerisch ermöglichen. All diesen Steuerelementen gemeinsam ist die 'VWItem'-Eigenschaft, die die darzustellende oder zu schreibende Prozessvariable auswählt. Das Verfahren ist also bei allen Steuerelemente gleich oder zumindest ähnlich. Um einen Überblick über die Funktionen eines Steuerelementes zu gewinnen, kann es auf ein Formular gezogen, dort selektiert und über die Taste "F1" die Hilfeseite aufgerufen werden.

2.2.5 Grundregeln Applikationsaufbau

Selten wird eine Visualisierung mit nur einer Bildschirmansicht auskommen. Üblich und nützlich ist die Unterteilung in unterschiedliche Bildschirmansichten nach Funktion oder Logik der Anlage. Ausgehend von einer zentralen Übersichtsseite kann der Benutzer dann über Schaltflächen in die ihn betreffenden Applikationsteile wechseln und dort seine Eingaben oder Beobachtungen vornehmen. Zusätzlich zu beachten dabei ist die personenbezogene Freigabe von Funktionen. Allgemein üblich ist hier z.B. der Zugriff auf eine Service-Seite, die nicht vom Bedienpersonal, aber von einem Techniker aufgerufen werden darf.

In der Praxis werden die unterschiedlichen Bildschirmansichten durch Formulare gestaltet. Formulare werden im Projektexplorer über den gleichnamigen Knoten "Formulare" hinzugefügt.



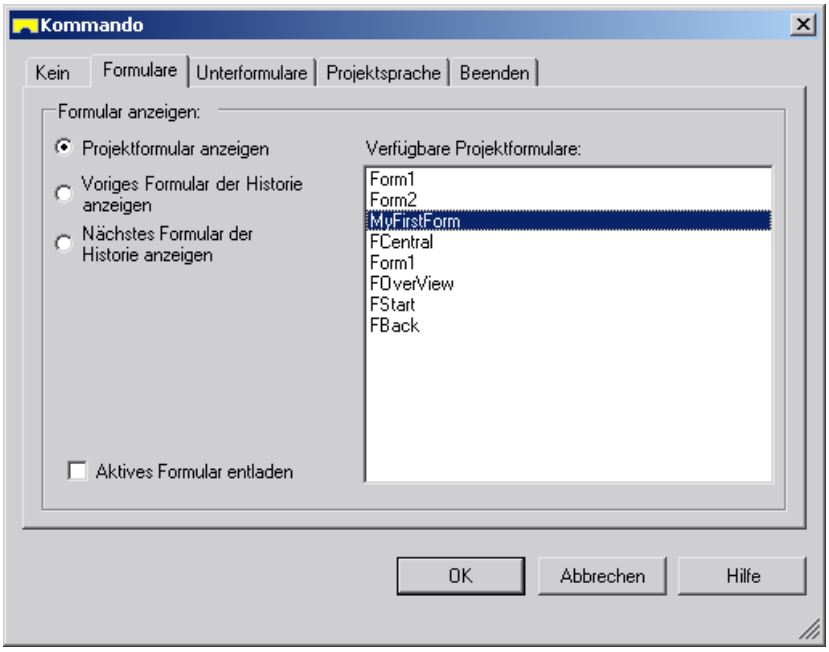
Ein neues Formular ist in ein Projekt einzufügen. Dazu ist der Knoten "Formulare" im Projektexplorer zu markieren und mit der rechten Maustaste das Kontextmenü aufzurufen. Im folgenden Dialog ist der Name "MyFirstForm" für das Formular anzugeben.

Ein neuer Knoten mit dem Namen "MyFirstForm" wird unter "Formulare" eingefügt. Ein entsprechender Formulardesigner wird geöffnet. Auf diesem Designer können wieder Steuerelemente platziert und parametriert werden.

Zu klären wäre jetzt noch die Frage, wie das neue Formular zur Anzeige gebracht wird.

Dazu ist zum ersten Formular der Anwendung zu wechseln. Auf diesem Formular ist ein Steuerelement vom Typ "CommandButton" zu platzieren.

Der Eigenschaftendialog der Eigenschaft "Command" des "CommandButton"-Steuerelementes ist zu öffnen. Auf der Karteikarte "Formulare" ist der Eintrag "MyFirstForm" in der Formularliste zu selektieren. Die Auswahl ist über die OK-Schaltfläche zu bestätigen.



Die hier festgelegte Funktion des "CommandButtons" bewirkt, dass zur Laufzeit das Formular "MyFirstForm" angezeigt wird.

Der Test über die -Schaltfläche zeigt, dass das Formular angezeigt wird, allerdings mit einer Titelzeile und den üblichen Funktionsschaltflächen darin.

Die gebräuchlichste Bildschirmansicht einer Visualisierungsapplikation ist ein Vollbild ohne Titelzeile.

Nach dem Beenden der Applikation ist das Formular "MyFirstForm" als Formulardesigner wieder sichtbar zu schalten.

Im Eigenschaftenfenster sind folgende Eigenschaften zu ändern:

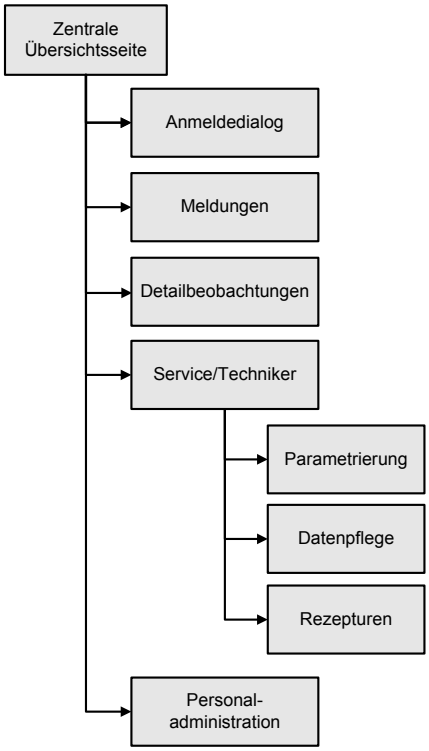
Eigenschaft	Wert
FormBorderStyle	None
WindowState	maximized

Da das Formular mit den hier eingestellten Eigenschaften jetzt keine Titelzeile mehr hat, kann es nicht ohne Hilfe einer Schaltfläche geschlossen werden. Auch kann es nicht mehr beiseite gerückt werden, da es maximiert den gesamten Bildschirminhalt füllt.

Auf dem Formular ist ein "CommandButton" zu platzieren. Die Eigenschaft "Command" ist so zu parametrieren, dass es wieder das erste Formular der Applikation aufruft.

Der Testlauf zeigt, dass es nun möglich ist, zwischen den beiden Formularen zu wechseln. Über den "CommandButton" sind also die einzelnen Seiten untereinander aufrufbar.

Eine wesentliche Überlegung beim Aufbau einer Visualisierung ist die Aufteilung der Bildschirmansichten. Ein typisches Beispiel soll hier gezeigt werden:



Jeder der hier dargestellten Funktionsblöcke soll durch eine oder mehrere Bildschirmansichten dargestellt werden. Eine so gezeichnete Hierarchie stellt die Grundzüge dar, wie die Seiten untereinander zu verknüpfen sind, welche Seiten also wo aufgerufen werden müssen. Wenn z.B. aus der zentralen Übersicht auf die einzelnen darunter liegenden Funktionsblöcke zugegriffen werden soll, sind mindestens 5 Schaltflächen nötig. Der Zugriff auf die Daten von Service/Techniker muss nicht auf der zentralen Übersichtsseite liegen, wenn eine Bildschirmansicht den Zugriff auf die Blöcke "Parametrierung", "Datenpflege" und "Rezepturen" enthält. Geschmackssache ist, ob parallel zueinanderliegende Bildschirmseiten untereinander verknüpft werden müssen. Eine Ausnahme hierbei ist jedoch mit Sicherheit eine Alarmansichtenseite, die auf jeden Fall schnell aus allen Teilen der Applikation erreichbar sein sollte.

2.2.6 Sprachumschaltung

Die Sprachumschaltung ermöglicht den Wechsel sämtlicher in der Applikation verwendeter Texte in Abhängigkeit von der gewählten Sprache. Sie wird verwendet um:

- Die Auslieferung einer international verwendbaren Applikation zu ermöglichen. Hier würde genügen, die anzuzeigende Sprache beim Projektstart oder auf einer Serviceseite zu konfigurieren.
- Internationalem Bedienpersonal die Bedienung zu ermöglichen. Denkbar ist hier, dass beispielsweise beim Schichtwechsel die Sprache umzustellen ist, da sich die Nationalität des Personal ändert.

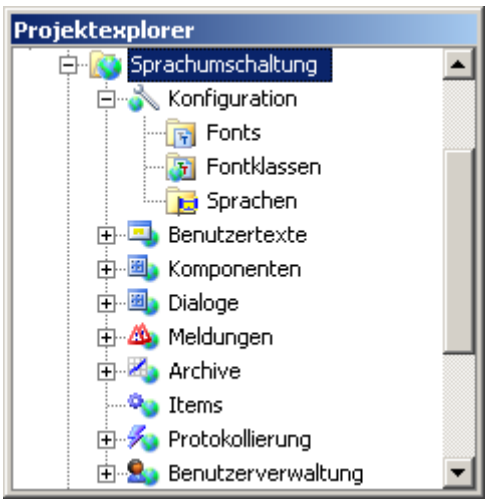
Beides ist quasi auf Knopfdruck möglich. Auch die Eingabe von nichtwestlichen Schriftzeichen ist über Unicode-Zeichensätze erlaubt.

Neben den eigentlichen Texten, die in der Applikation verwendet werden, sind weitere Besonderheiten zu beachten:

- In Abhängigkeit von der gewählten Sprache müssen Einheiten (z.B. Temperatur °C/°F) umgerechnet und umgestellt werden
- In Abhängigkeit von der gewählten Sprache müssen Datums- und Zeitformate im landesüblichen Format dargestellt werden.
- Gegebenenfalls ist die Größe von Schriftarten in unterschiedlichen Zeichensätzen anders normiert, so dass beim Wechsel der Sprache eine Anpassung der Schriftart nötig wird.

Die Angabe der Texte erfolgt in der Sprachumschaltung von VisiWinNET. Die Anbindung an die Steuerelemente erfolgt über verschiedene Eigenschaften.

Der Knoten "Sprachumschaltung" im Projektexplorer ist zu expandieren.



Die Funktionen der Sprachumschaltung ist in verschiedene Bereich gegliedert:

- "Benutzertexte" sind die Texte, die der Projekteur anlegen kann, um sie in der Applikation anzuzeigen.
- "Komponenten" und "Dialoge" enthalten die Texte, die von den Steuerelementen und Dialogen des VisiWin-Pakets verwendet und erwartet werden. Hier können keine Texte angelegt werden, die Texte können sprachlich erweitert oder applikationsbezogen geändert werden.
- in den Knoten "Variablen", "Alarme", "Archive", Benutzerverwaltung" und "Protokollierung" können keine Texte angelegt werden. Das wird von den Editoren der entsprechenden Systeme erledigt. Wird beispielsweise ein Alarm im Alarmsystem angelegt, erscheint der sprachumschaltbare Text des Alarms unter dem hier angezeigten Knoten. Dies entspricht dem zentralen Verwaltungsgedanken der Sprachumschaltung.

Der "Konfiguration"-Knoten beinhaltet folgende Untergruppen

- "Fontklassen" und "Fonts" definieren die Funktion der Schriftartenumschaltung
- "Sprachen" beinhaltet als Liste die Projektsprachen. Ein neues Projekt beinhaltet die Sprachen Deutsch und Englisch. Weitere Sprachen können dem Projekt hinzugefügt werden.

Die Einheitenumschaltung ist im Variablenystem enthalten. Sie ist konzeptionell von der Sprachumschaltung getrennt worden, da die zu definierenden Informationen als Hauptmerkmal Umrechnungswerte enthalten.

Der Knoten "Benutzertexte" ist zu markieren. Über das Kontextmenü ist eine neue Textgruppe anzulegen. Die Textgruppe ist über die VisiWinNET Eigenschaftenseite umzubenennen in "MyFirstForm".

Sprachumschaltbare Texte können in einzelnen Gruppen angelegt und verwaltet werden. Dies hat folgende Vorteile:

- Über den Namen der Textgruppe kann angegeben werden, wo die Texte verwendet werden. Über Textgruppen kann also eine Gliederung erfolgen.
- Die Texte, die angelegt und später vielleicht einmal geändert werden sollen, können leichter wiedergefunden werden.
- Wird zu jedem Formular eine Textgruppe angelegt, erleichtert dies später die Arbeit. Wenn das Formular in ein anderes Projekt übernommen wird, muss nur die entsprechende Textgruppe in das Projekt kopiert werden. Dann ist das Formular, das die Texte verwendet, wieder funktionsfähig.

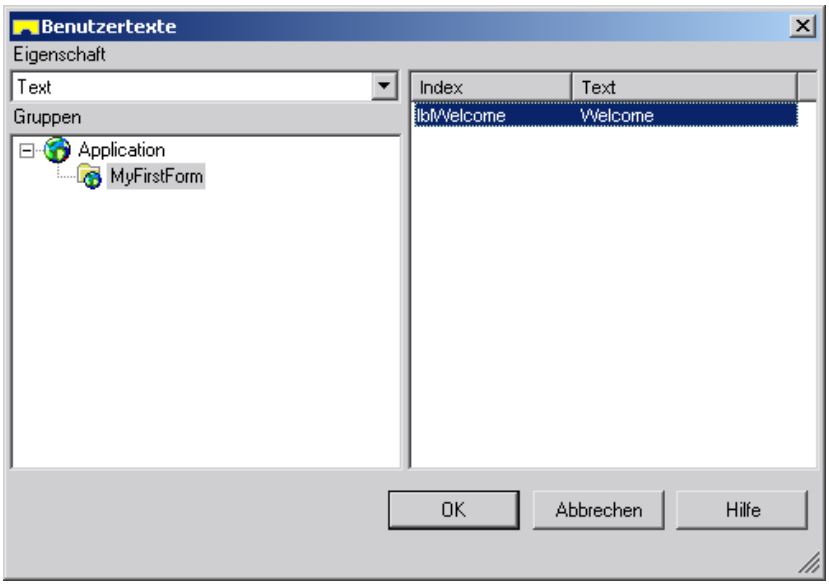
Im Tabelleneditor ist in der "Texte"-Tabelle über die Taste "F8" ein neuer Text anzulegen. Der Text ist folgendermaßen anzupassen:

Parameter	Wert
Name	lblWelcome
Deutsch (Deutschland) (1031)	Willkommen
Englisch (USA) (1033)	Welcome

Über den entsprechenden Knoten unter der Gruppe "Formulare" im Projektexplorer ist das Formular "MyFirstForm" zu öffnen. Hier sind ein 'Label'- und zwei 'CommandButton'-Steuerelemente zu platzieren.

Das 'Label'-Steuerelement ist zu markieren. Über die -Schaltfläche der 'LocalizedText'-Eigenschaft ist der Dialog zur Auswahl eines sprachumschaltbaren Textes zu öffnen.

Hier werden alle im Projekt angelegten Benutzertexte zur Auswahl angeboten.



In der Textgruppenansicht (links) ist zur Textgruppe 'MyFirstForm' zu navigieren. In der Textauswahlansicht (rechts) ist der Text "lblWelcome" zu markieren. Die Auswahl ist über "OK" zu bestätigen.

Über die Auswahl wird festgelegt, welcher Text vom Steuerelement zur Laufzeit angezeigt werden soll. Nach der Übernahmen wird im Steuerelement der Text in der aktiven Sprache angezeigt.

Die beiden 'CommandButton'-Steuerelemente sollen dazu dienen, die Sprache zur Laufzeit umzuschalten.

Das erste 'CommandButton'-Steuerelement ist zu markieren. Über die ...-Schaltfläche der 'Command'-Eigenschaft ist der Dialog zur Funktionsauswahl zu öffnen. Auf der Karteikarte "Projektsprache" ist der Eintrag "Deutsch (Deutschland)" auszuwählen. Die Auswahl ist über die "OK"-Schaltfläche zu bestätigen.

Für das zweite 'CommandButton'-Steuerelement ist der Vorgang zu wiederholen. Dabei ist die Sprache "Englisch (USA)" auszuwählen.

Als Eigenschaftenwerte werden die sogenannten "Locale Identifier" (LCID) angezeigt. Jede Sprache hat unter Windows einen solchen Schlüssel. Für Deutsch wird hier "1031" angezeigt, für Englisch "1033".

Der Testlauf zeigt, dass über die Schaltflächen die Sprache in dem 'Label'-Steuerelement umgeschaltet wird.

Texte sind in ihrem Sprachumfang erweiterbar. Neue Projektsprachen werden über den Knoten "Sprachumschaltung→Konfiguration→Sprachen" eingefügt.

Über das Einfügen einer neuen Projektsprache werden die Texttabellen um jeweils eine Spalte erweitert.

Im Projektexplorer ist der Knoten "Sprachumschaltung→Konfiguration→Sprachen" zu markieren. Im Tabelleneditor ist eine neue Sprache mit folgenden Parametern einzufügen:

Parameter	Wert
Sprache	Dänisch (Dänemark)
Eingabegebiets-schema	Dänisch (Dänemark) | Dänisch (Dänemark)

Das genannte Eingabegebietsschema ist nicht vorhanden!?!

Eingabegebietsschemata sind Unterstützungen des Betriebssystems, die die länderspezifischen Schrift-/Sonderzeichen auf der Tastatur umschalten. So beinhaltet z.B. eine Deutsche Tastatur die Umlaute ä, ö, und ü, die im englischen nicht existieren. Auf englischen Tastaturen sind diese Tasten mit anderen Zeichen belegt. Der gesendete Tastencode der Tasten ist jedoch identisch. Das Betriebssystem regelt die Zuordnung der Tasten zu den auf der Tastatur aufgedruckten Zeichen über eine länderspezifische Tabelle, dem "Eingabegebietsschema".

Benötigte Eingabegebietsschemata müssen installiert werden. dazu ist in die Systemsteuerung des Rechners zu wechseln. (Windows-Startmenü→Einstellungen→Regions- und Sprachoptionen→Karteikarte "Sprachen"→Schaltfläche "Details"). Hier ist die Sprache "Dänisch" hinzuzufügen. Gegebenenfalls sind dazu auf Anfrage Komponenten von der Windows-CD nachzuinstallieren.

Danach ist das Eingabegebietsschema auch in der Auswahlliste des Tabelleneditorfeldes vorhanden.

Das Muppet Labor: Die folgende Einstellung zeigen Sie bitte keinem Dänen.

In der dänischen Spalte des Textes 'lblWelcome' ist das Wort "Smärebräd" einzugeben.

Das "ä" wird dabei im Editor durch das dänisch-typische Schriftzeichen ersetzt. Die Einstellung eines Eingabegebietsschemas ermöglicht also Zeichen, die sonst über die Tastatur nicht zu erreichen sind.

Um das Beispiel hier zu komplettieren, ist auf dem Formular "MyFirstForm" ein drittes 'CommandButton'-Steuerelement zu platzieren und über die 'Command'-Eigenschaft auf die neue Sprache "Dänisch" zu konfigurieren.

Wenn beispielsweise für die dänische Sprache eine besondere Darstellung bezüglich der Schriftart nötig wird, kann dies mithilfe von den Definitionstypen "Fonts" und "Fontklassen" erfolgen.

Im Projektexplorer ist der Knoten "Konfiguration→Fontklassen" zu markieren

Die im Tabelleneditor dargestellten Fontklassen haben einen Namen und beinhalten für jede Projektsprache eine Schriftart.

In der Fonklassentabelle ist durch das Hinzufügen einer Sprache ebenfalls eine neue Spalte "Dänisch" hinzugefügt worden. Die Werte sind nicht ausgefüllt, können jedoch mit den Fontdefinitionen belegt werden.

Ist ein Steuerelement mit einer Fontklasse verknüpft, wechselt es bei einer Sprachumschaltung zu der entsprechenden Schriftart.

Das Feld "Dänisch" der Fontklasse "Labels" ist zu markieren. In der Auswahlliste des Feldes ist die Schriftart "Tahoma8BoldWestern" auszuwählen.

Auf dem Formular "MyFirstForm" ist das 'Label'-Steuerelement zu markieren. Über die ...-Schaltfläche der 'FontClass'-Eigenschaft ist der Dialog zur Auswahl einer Fontklassendefinition zu öffnen. Hier ist die Fontklasse "Labels" auszuwählen. Das Projekt ist zu starten.

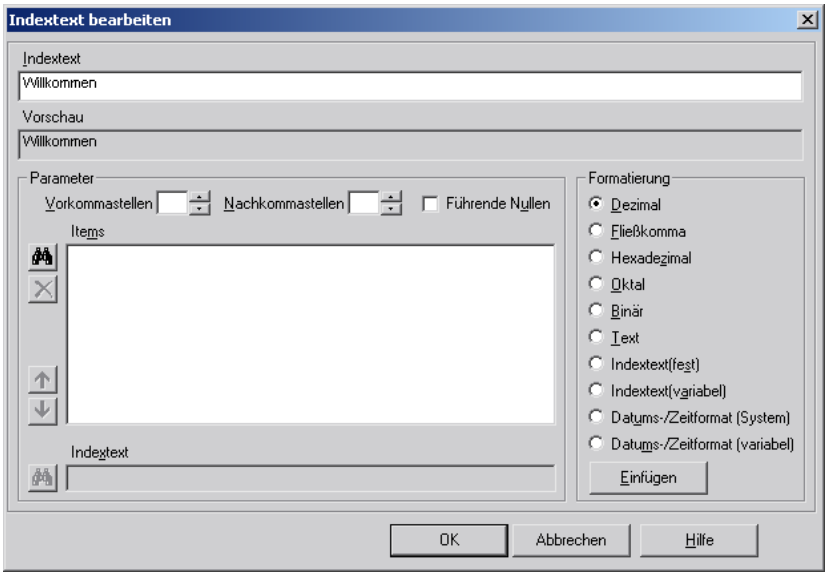
Der Testlauf zeigt, dass sich nun auch die Schriftart ändert, wenn zur dänischen Sprache gewechselt wird.

Ein weiteres Feature ist das dynamische Einbinden von Zusatzinformationen in Texte:

- Prozessvariablenwerte
- Andere Texte
- Datums-/Zeitformate

Zur Entwicklungszeit werden durch "@"-Zeichen umfasste Formatzeichenketten in den Text eingefügt. Zur Laufzeit werden diese Bereiche mit den entsprechenden Werten ersetzt.

Der Text "lblWelcome" ist im Tabelleneditor sichtbar zu schalten. Über die ...-Schaltfläche in der Spalte "Deutsch" ist der Dialog zum Bearbeiten des Textes zu öffnen.



Dieser Dialog enthält einen Formatzeichenkettengenerator, dessen Funktionen das Einfügen der o.g. Funktionen erlaubt.

Über die -Schaltfläche links neben der "Variablen"-Liste ist der Variablenauswahldialog zu öffnen. Hier ist eine Variable auszuwählen.

In der Formatierungsliste ist der Eintrag "Dezimal" zu wählen. Über die "Einfügen"-Schaltfläche ist die Formatzeichenkette in den Text einzufügen.

Nachdem die Einstellungen über die OK-Schaltfläche bestätigt wurden, steht im Textfeld folgender Text:

"Willkommen @1d@"

Die in "@"-Zeichen eingefasste Formatierungszeichenkette gibt an, dass der erste in der Prozessvariablenliste angegebene Prozesswert als Dezimalwert an dieser Stelle ausgegeben werden soll.

Der Testlauf zeigt, dass tatsächlich der entsprechende Wert der Variablen im Text eingeblendet wird.

Neben dem 'Label'-Steuerelement unterstützen auch andere Steuerelemente direkt oder indirekt die Anzeige von sprachumschaltbaren Texten. Generell gilt: Eine 'LocalizedText'-Eigenschaft kennzeichnet die Anzeige eines sprachumschaltbaren Textes.

Andere Steuerelemente haben keine solche Eigenschaft, beziehen aber Texte oder Formatierungsanweisungen aus der Sprachumschaltung. Das 'AlarmLine'-Steuerelement beispielsweise beinhaltet die Eigenschaft 'DateTimeFormat', die die Angabe eines Textes aus der Sprachumschaltung erlaubt. Die Angabe "@LongDate" als Eigenschaftenwert besagt, dass der gleichnamige Text aus der Textgruppe "Komponenten.Time.DateFormats" für jede Projektsprache eine Formatierungszeichenkette enthält, die zur Laufzeit dazu verwendet wird, Datums-/Zeitwerte zu formatieren. Auch dies ist eine Funktion der Sprachumschaltung.

2.2.7 Alarmsystem

Das Alarmsystem dient zur zentralen Erfassung und Speicherung von Fehlerzuständen oder spezifischen Alarmen. Die entsprechenden Definitionen, welche Variable einen Alarm auslöst, wie der Alarm dargestellt wird und welche Zusatzinformationen zum Alarm gehören, werden im Alarmeditor projektiert. Die Variablen, die die Alarme auslösen, werden von VisiWinNET beim Variablenkern zur Überwachung angemeldet. Bei einer Variablenwertänderung wird der entsprechende Alarm zur Applikation übertragen, wo sie in speziellen Steuerelementen angezeigt wird.

VisiWinNET beinhaltet zwei Steuerelemente, die die direkte Anzeige anstehender Alarme erlauben:

AlarmList Alle anstehenden Alarme werden in Form einer Liste angezeigt. Das Steuerelement unterstützt dabei zusätzlich die Quittierung von Alarmen.

AlarmLine Die platzsparende Variante, die auf allen Formularen im Kopf- oder Fußbereich Einsicht in anstehende Alarme erlaubt.

Zusätzlich zur Übertragung der Alarmdaten in die Oberfläche erlaubt VisiWinNET auch die Speicherung von Alarmen in Dateien. Dadurch ist ein Rückblick auf den Produktionsverlauf einer Anlage möglich. Das Steuerelement 'HistoricalAlarmList' dient zur Anzeige dieser aufgezeichnete Daten in Form einer Liste.

Ein Alarm wird durch die Änderung eines Bits aus dem Variablenkern gesteuert. Wählbar ist, ob das Bit auf "1" oder "0" gesetzt wird, damit der Alarm ausgelöst wird. Über eine Erweiterung ist es aber auch möglich, analoge Größen beispielsweise auf Grenzwertüberschreitungen zu überprüfen. Alarme können vom Benutzer quittiert werden. Eine Quittierrückmeldung an die Steuerung ist ebenfalls realisierbar.

Das Erscheinungsbild von Alarmen in den Steuerelementen 'AlarmLine' und 'AlarmList' wird über Alarmklassen parametriert. Alarmklassen beinhalten verschiedene Parameter (Farben, Statustexte und Symbole), die die Anzeige eines Alarms beschreiben. Jeder Alarm referenziert eine Alarmklasse. So stehen jedem Alarm die Anzeigeparameter einer Alarmklasse zur Verfügung.

Alarmgruppen gliedern Alarme. Ähnlich wie in der Sprachumschaltung ist es sinnvoll, die einzelnen funktionalen Elemente einer Anlage zu gruppieren. Dies fördert die Übersichtlichkeit und macht das Projekt modularer. Alarmgruppen beinhalten aber auch zusätzlich Parameter, die die gruppenweise Deaktivierung oder Quittierung von Alarmen zulassen.

Das folgende Beispiel zeigt:

- Das generelle Projektieren, Auslösen und Anzeigen von Alarmen, die als Bits im Variablenkern vorliegen
- Das Quittieren von Alarmen
- Verschiedene Anzeigefilter
- Die Überwachung von analogen Werten

Als erstes sind die Quellen der Alarme zu definieren. Da Alarme variablengebunden sind, beginnt also alles in der Prozessanbindung. Da allerdings keine realen Steuerungsdaten zur Verfügung stehen, müssen die auslösenden Variablen über interne Variablen simuliert werden.

Im Projektexplorer ist der Knoten "Variablen" zu expandieren. Über den Knoten "Kanäle→Intern" ist der Tabelleneditor zu öffnen. Hier sind folgende Variablen zu definieren:

Name	Datentyp
MessageSource	VT_I2
MessageState	VT_I2
GroupFunction	VT_I2

Als funktionale Datenanbindung reichen die hier definierten Variablen:

- MessageSource beinhaltet die Bits, die die Alarme auslösen
- MessageStates beinhaltet die von der Kernfunktionalität zurückgegebenen Alarmzustände
- GroupFunction dient zur Steuerung von Alarmfunktionalitäten über die übergeordnete Gruppe

Nun folgt die Festlegung der Definitionen im Alarmsystem:

Der Knoten "Alarme" im Projektexplorer ist zu expandieren. Das Kontextmenü des darunter liegenden Knotens "Alarmgruppen" ist über einen Klick mit der rechten Maustaste aufzurufen. Über den Eintrag "Neu" ist eine Alarmgruppe zu erzeugen. In der Alarmgruppe sind folgende Parameter einzustellen:

Parameter	Wert
Name	Grp1
Quittierungsvariable	GroupFunction
Bitnummer Quittierungsvariable	0

Eine zweite Alarmgruppe ist unter dem Knoten "Alarmgruppen" zu erzeugen

Vorsicht: Alarmgruppen sind schachtelbar. Daher zuerst zum Knoten "Alarmgruppen" wechseln

Die Parameter der zweiten Alarmgruppe sind auf folgende Werte einzustellen:

Parameter	Wert
Name	Grp2
Quittierungsvariable	GroupFunction
Bitnummer Quittierungsvariable	1

Im Tabelleneditor sind in der Tabelle "Alarme" zwei neue Alarme anzulegen. Nachdem die Eingabemarke auf einen Datensatz im Tabelleneditor gesetzt wurde, werden in der VisiWinNET-Eigenschaftenseite die Parameter der Alarmdefinitionen dargestellt. Folgende Einstellungen sind hier vorzunehmen:

Für den 1. Alarm

Parameter	Wert
Name	Msg1
Ereignisvariable	MessageSource
Bitnummer	0
Alarmtext	MessageSource Bit 0
Priorität	1
Statusvariable	MessageState
Bitnummer Statusvariable	0

Für den 2. Alarm

Parameter	Wert
Name	Msg2
Ereignisvariable	MessageSource
Bitnummer	1
Alarmtext	MessageSource Bit 1
Priorität	2
Statusvariable	MessageState
Bitnummer Statusvariable	4

Für die Auswahl von Variablen steht in der VisiWinNET-Eigenschaftenseite die -Schaltfläche rechts neben dem entsprechenden Parameterfeld zur Verfügung.

Zwei weitere Alarme sind unter "Grp2" anzulegen. Auf die gleiche Weise wie oben sind die Parameter dieser Alarme anzupassen:

Für den 1. Alarm

Parameter	Wert
Name	Msg3
Ereignisvariable	MessageSource
Bitnummer	2
Alarmtext	MessageSource Bit 2
Priorität	3
Statusvariable	MessageState
Bitnummer Statusvariable	8

Für den 2. Alarm

Parameter	Wert
Name	Msg4
Ereignisvariable	MessageSource
Bitnummer	3
Alarmtext	MessageSource Bit 3
Priorität	4
Statusvariable	MessageState
Bitnummer Statusvariable	12

Die oben angegebenen Werte werden im Folgenden die volle Funktion des Alarmsystems zeigen. Der Parameter "Statusvariable" ist optional. Die Festlegung eines Alarmtextes ist quasi Pflicht. Die Angabe der Ereignisvariable schließlich muss sein.

Alle neu angelegten Alarme zeigen mit dem Parameter "Alarmklasse" auf den Wert "Alarm". Dahinter steckt eine Definition, die sich unter dem Knoten "Alarmklassen" des Projektexplorers verbirgt.

Der Knoten "Alarmklassen" im Projektexplorer ist zu markieren. Im Tabelleneditor ist die Eingabemarke auf die "Alarm"-Alarmklasse zu setzen. In der VisiWinNET-Eigenschaftenseite ist der Parameter "Quittierungsart" auf "quittierpflichtig" zu setzen.

VisiWinNET unterstützt verschiedene Quittierungsarten. Eine Quittierungsart legt fest, wie oft ein Alarm vom Benutzer oder von der Steuerung quittiert werden muss, bevor sie aus den Steuerelementen der Applikation verschwindet. Durch diese Einstellung kann ein Alarm zur Laufzeit verschiedene Stati einnehmen. Beispielsweise unterscheidet das System zwischen dem Zustand "gekommen" (Ereignisbit wurde gesetzt) und "gekommen quittiert" (Nach dem Setzen wurde vom Benutzer das Auftreten quittiert, das Ereignisbit ist aber noch nicht zurückgesetzt worden). Jeder Zustand kann durch ein Symbol, einen Statustext und Farben kenntlich gemacht werden. Auch diese Festlegungen erfolgen in den Alarmklassen.

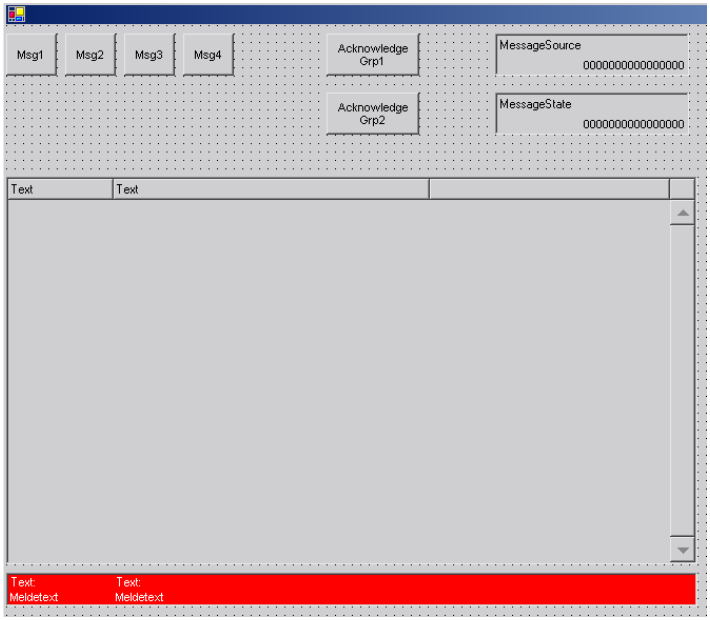
Der Status eines Alarms kann auch bitcodiert in einer Variablen hinterlegt werden. Dazu sind in den Alarmen die Statusvariablen mit Bitnummern festgelegt worden. Jeweils drei Bits repräsentieren den Status eines Alarms.

Damit auch alle definierten Information in der Anzeige erscheinen, müssen nun die Oberflächenelemente projektiert werden. Der Übersicht halber erfolgt dies in einem eigenen Formular.

Über den Knoten "Formulare" im Projektexplorer ist ein neues Formular "MyAlarmForm" anzulegen. Mithilfe von "CommandButton"-Steuerelementen sind die entsprechenden Seitenwechsel zu dem Formular zu implementieren.

Auf diesem Formular sind folgende Steuerelemente (von oben nach unten, siehe Grafik) zu platzieren:

- 2 x VarOut
- 4 x Switch
- 2 x Key
- 1 x AlarmList
- 1 x AlarmLine



Die vier 'Switch'-Steuerelemente sind über die 'VWItem'-Eigenschaft mit der Variablen "MessageSource" zu verbinden. Die 'BitNumber'-Eigenschaft im VWItem-Objekt ist dabei für die einzelnen Steuerelemente auf die Werte 0, 1, 2 und 3 zu setzen. Dies sind die Bits, die die Alarme auslösen. Über die Text-Eigenschaft kann die Funktion der Schaltfläche angegeben werden.

Die beiden 'Key'-Steuerelemente sind über die 'VWItem'-Eigenschaft an die Variable "GroupFunction" anzubinden. Die 'BitNumber'-Eigenschaft im 'VWItem'-Objekt ist dabei für das zweite Steuerelement auf den Wert 1 zu setzen. Die beiden angeschlossenen Bits sind die Quittierbits der Alarmgruppen. Auch hier kann über die 'Text'-Eigenschaft die Funktion der Schaltflächen abgebildet werden.

Die zwei 'VarOut'-Steuerelemente sind über die VWItem-Eigenschaft mit den beiden internen Variablen "MessageSource" und "MessageState" zu verknüpfen. Zusätzlich ist die 'Format'-Eigenschaft auf 'binary' einzustellen. Über die Eigenschaft Label→Text→Text ist der angebundene Variablename als Informationstext einzugeben.

Die Eigenschaften der Steuerelemente 'AlarmList' und 'AlarmLine' müssen für den ersten Testlauf nicht angepasst werden.

Die Applikation ist zu starten. Die vier unteren Bits der Variablen "MessageSource" sind über die entsprechenden 'Switch'-Steuerelemente zu setzen.

Folgende Änderungen werden sichtbar:

- In der Alarmliste werden alle vier Alarme angezeigt
- In der Alarmzeile wird der Alarm "Msg4" angezeigt. Dies ist der aktuell im System anstehende Alarm mit der höchsten Priorität
- Die Variable "MessageState" zeigt jeweils für jeden Alarm den Status "100" (binär) an. Nötigenfalls könnte dieser Status auch an die Steuerung zurück übertragen werden.

Die vier unteren Bits der Variablen "MessageSource" sind über einen erneuten Klick auf die 'Switch'-Steuerelemente wieder auf "0" zu setzen.

- Die Alarmzeile zeigt keine Alarme mehr
- In der Alarmliste bleiben die Alarme erhalten. Es ändert sich jedoch der Status und damit die Farbe. Durch die gewählte Quittierungsart ist eine Quittierung nötig, um die Alarme endgültig zurückzusetzen.
- Die Statusbits von Alarmen und Alarmgruppen zeigen jetzt den Status "010"

Die Alarme sind nacheinander durch einen Doppelklick auf die entsprechenden Einträge in der Alarmliste zu quittieren.

- Die Alarme verschwinden aus der Alarmliste.
- Auch die Statusbits werden in "MessageState" zurückgesetzt

Die Alarme sind erneut über die 'Switch'-Steuerelemente zu generieren.

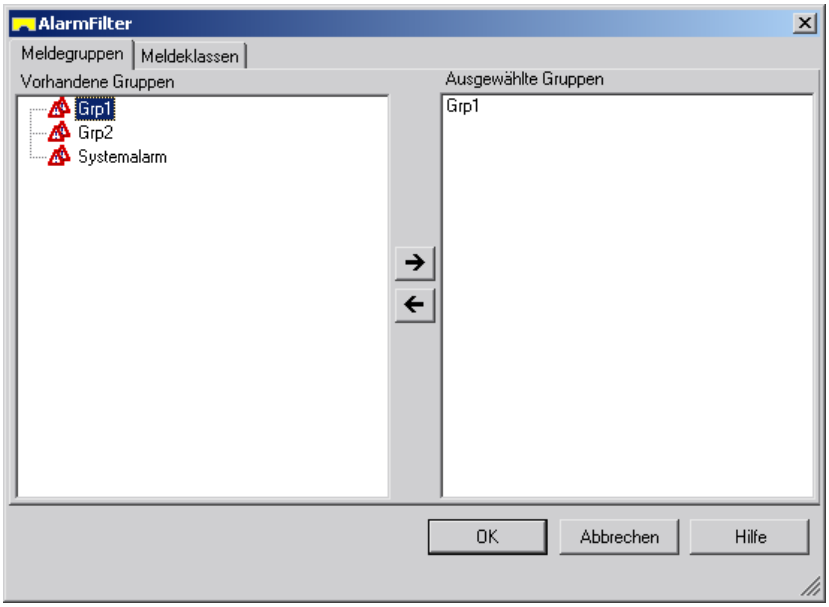
Die beiden 'Key'-Steuerelemente sind zu drücken.

Über das Quittierbit einer Gruppe werden die enthaltenen Alarme gemeinsam quittiert.

Hier kann das Projekt gestoppt werden.

VisiWinNET enthält die Möglichkeit, die Anzeige von Alarmen in den Steuerelementen zu filtern. Dadurch ist es beispielsweise möglich, eine Alarmliste nur mit Alarmen über Anlagenstörungen zu projektieren und eine weitere mit Produktionshinweisen. Im Client-Server-System wird dadurch auch möglich, nur die für den Arbeitsplatz wichtigen Alarme anzuzeigen und trotzdem eine zentrale Datenhaltung zu behalten.

Zurück in der Designansicht ist das 'AlarmList'-Steuerelement zu markieren. Über die -Schaltfläche der 'AlarmFilter'-Eigenschaft ist der Dialog zur Filterfestlegung zu öffnen.



Der Filter erlaubt die Auswahl von Alarmgruppen und Alarmklassen. Nur die Alarme, die in den hier ausgewählte Gruppen liegen, werden im Steuerelement angezeigt.

Über einen Doppelklick auf den Eintrag "Grp1" ist die gleichnamige Gruppe auszuwählen. Nach Bestätigen der Auwahl über die OK-Schaltfläche ist das Projekt erneut zu starten. Noch einmal sind alle vier Alarme über "MessageSource"zu generieren.

Nur die in "Grp1" liegenden Alarme werden in der Alarmliste angezeigt. Die Statusbits zeigen aber, dass auch die in "Grp2" liegenden Alarme generiert wurden. Der Filter beschränkt sich also wirklich nur auf die Anzeige im Steuerelement.

Nachdem also hoffentlich Ursache und Wirkung von Alarmen hinreichend geklärt sind, folgt nun die Beschreibung der Analogwertüberwachung. Ein typisches Beispiel ist eine Temperatursensor, dessen Wert in unterschiedliche Bereiche (Normal, Kritisch, Fehler) eingeteilt wird.

Die folgende Beschreibung kann nicht für "Compact"- oder "Embedded"-Projekte ausgeführt werden.

Im Projektexplorer ist der Knoten "Projektkonfiguration→Serverkomponenten" zu markieren. In der VisiWinNET-Eigenschaftenseite ist der Eintrag "Supervision/Prozessdatenüberwachung" über das Kontrollkästchen zu aktivieren.

Im Projektexplorer wird ein neuer Knoten mit der Bezeichnung "Prozessdatenüberwachung" sichtbar.

Die Vorgehensweise zeigt, dass VisiWinNET aus einzelnen Komponenten besteht, die individuell in einem Projekt aktiviert oder deaktiviert werden können. Eine deaktivierte Komponente wird zur Laufzeit nicht geladen.

Zwei neue interne Variablen mit dem Datentyp "VT_I2" sind im Variableneditor anzulegen. Die Namen der beiden Variablen sollen "AnalogValue" und "SupervisionOut" sein.

Im Editor des Alarmsystems ist eine neue Alarmgruppe mit dem Namen "AnalogValueAlarm" anzulegen. In dieser Gruppe sind drei Alarme zu definieren, die folgende Parameterwerte haben:

Parameter	Alarm 1	Alarm 2	Alarm 3
Name	Msg1	Msg2	Msg3
Text	AnalogValue Warnung 1	AnalogValue Warnung 2	AnalogValue Alarm
Ereignis-variable	SupervisionOut	SupervisionOut	SupervisionOut
Bitnummer Ereignis-variable	3	4	5

Über das Kontextmenü des Knoten "Prozessdatenüberwachung→Überwachungsgruppen" ist eine neue Überwachungsgruppe anzulegen. Auf dem Dialog, der die Parameter der Überwachungsgruppe beinhaltet, sind folgende Einstellungen vorzunehmen:

Parameter	Wert
Name	Analog
Typ	Bereichsüberwachung

Im Tabelleneditor der Prozessdatenüberwachung ist in der Tabelle "Überwachungen" über die Taste "F8" eine neue Überwachung hinzuzufügen. Die Parameter dieser Überwachung sind mit folgenden Werten zu füllen:

Parameter	Wert
Name	AnalogValueSupervision
Überwachungs-variable	AnalogValue
Ausgabevariable	SupervisionOut
Bitnummer Ausgabevariable	0
Grenze HHH	100
Grenze HH	90
Grenze H	70

Das Formular "MyFirstAlarm" ist zu öffnen. Hier ist ein VarIn-Steuerelement hinzuzufügen und über die 'VWItem'-Eigenschaft mit der Variablen "AnalogValue" zu verbinden.

Zuletzt ist der Gruppenfilter der Alarmliste, der in den vorhergehenden Schritten eingestellt wurde, wieder zu entfernen (alle Gruppen wieder rausnehmen).

Das Projekt ist zu starten. Der Wert der "AnalogValue"-Variablen ist über das entsprechende 'VarIn'-Steuerelement auf die in der Überwachung projektierten Grenzwerte "70/90/100" zu setzen.

Die entsprechenden Alarme aus dem Alarmsystem werden ausgelöst. Konzeptionell sind überwachte Analogwerte also keine eigenständigen Alarme. Die Prozessdatenüberwachung prüft nur Prozesswerte auf Grenzwertüberschreitung und setzt entsprechend Bits in anderen Prozesswerten. Diese dienen dann dazu, bitgesteuerte Alarme auszulösen.

Vorsicht Falle: Sollten die Alarme in der Alarmliste nicht angezeigt werden, liegt das an dem im vorhergehenden Schritt festgelegten Gruppenfilter.

In der Alarmzeile sollte aber auf jeden Fall der Alarm sichtbar sein.

2.2.8 Archivsystem

Das VisiWinNET Archivsystem erlaubt die Aufzeichnung von Prozesswerten. Als Ansichtskomponente steht das 'TrendChart'-Steuerelement zur Verfügung. Die nötigen Definitionen, welche Prozesswerte wie aufgezeichnet werden sollen, werden im Archiveditor vorgenommen.

Im Projektexplorer ist der Knoten "Trends→Archive" zu markieren. Über das Kontextmenü ist eine neue Archivdefinition anzulegen. Über die VisiWinNET-Eigenschaftenseite sind folgende Parameter der neuen Definition anzupassen:

Parameter	Wert
Name	Archiv1
Typ (Karteikarte "Format")	Folgearchiv
Abtastzeit (Karteikarte "Aufzeichnung")	10

Ein Archiv ist ein Container für aufzuzeichnende Werte Dieser Container ...

- ... fasst mehrere aufgezeichnete Prozesswerte in Dateien zusammen
- ... legt das Abtastverhalten fest
- ... administriert die erzeugten Dateien

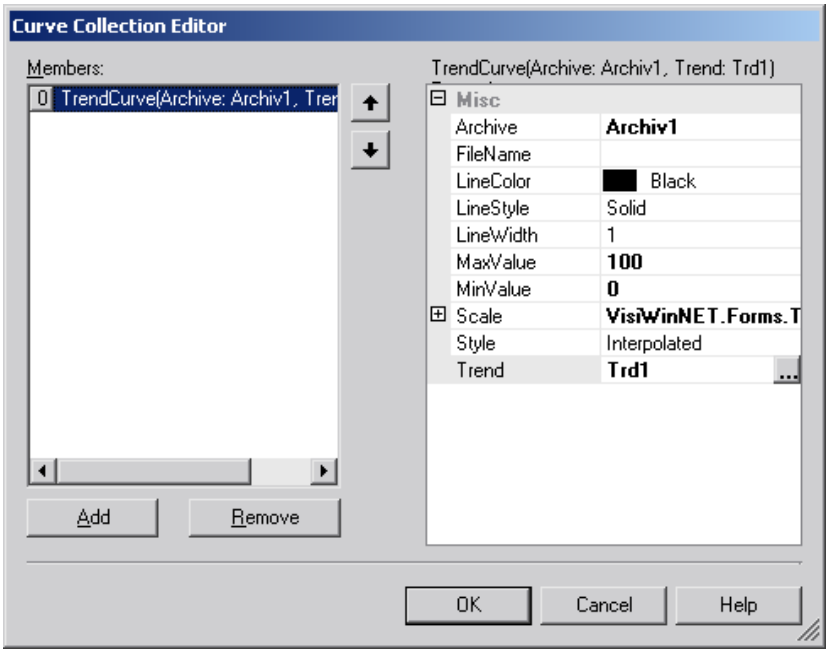
Die Festlegung der aufzuzeichnenden Prozesswerte wird im folgenden Schritt beschrieben.

Im Tabelleneditor ist über "F8" eine neue Trenddefinition anzulegen. Über die -Schaltfläche in der Spalte "Trendvariable" ist der Variablenauswahldialog zu öffnen. Ist in den vorhergehenden Schritten ein OPC-Server verwendet worden, ist die Variable "Ch1.DistillationDevice.Unit1.WaterTank.Temperature" auszuwählen.

Wurde in den vorhergehenden Schritten ein VisiWin-Treiber verwendet, kann hier die Variable "Ch1.DummyItem<.Saege1>" verwendet werden.

Damit ist auch die Festlegung des Prozesswertes erledigt. Es fehlt noch die Ansicht der aufgezeichneten Werte in der Applikation.

Ein neues Formular mit dem Namen "MyFirstTrend" ist zu erzeugen und über entsprechend parametrierte 'CommandButton'-Steuerelemente aus den anderen Formularen aufrufbar zu machen. Auf diesem Formular ist ein Steuerelement vom Typ 'TrendChart' zu platzieren. Über die -Schaltfläche der 'Curves'-Eigenschaft ist der Dialog zur Parametrierung der anzuzeigenden Kurven zu öffnen.



Die Liste auf der linken Seite enthält die zu zeichnenden Kurven. Eine Kurve ist schon im Dialog eingefügt, allerdings hat sie noch keine Anbindung an eine Trenddefinition.

Der Eintrag links ist zu markieren. Über die [...]-Schaltfläche der 'Archive'-Eigenschaft rechts ist der Dialog zur Auswahl von Archiv- und Trenddefinitionen zu öffnen. In der "Archiv"-Liste ist "Archiv1" auszuwählen und in der Trendliste ist "Trd1" zu markieren. Die Auswahl ist über die "OK"-Schaltfläche zu bestätigen. Anschließend ist die Eigenschaft 'MaxValue' auf 150 einzustellen.

Die Dialogauswahl wird in die Eigenschaften 'Archive' und 'Trend' übernommen. 'MaxValue' parametriert die obere sichtbare Wertegrenze für diese Kurve.

Der Dialog ist über "OK" zu schließen. Die Eigenschaft 'TimeScale' ist zu expandieren. In der darunterliegenden 'Resolutiuon'-Eigenschaft ist der Wert "00:02:00" anzugeben. Das Projekt ist zu starten.

Die 'Resolution'-Eigenschaft legt den sichtbaren zeitlichen Bereich im Steuerelement fest. Hierüber kann also die x-Achse parametriert werden.

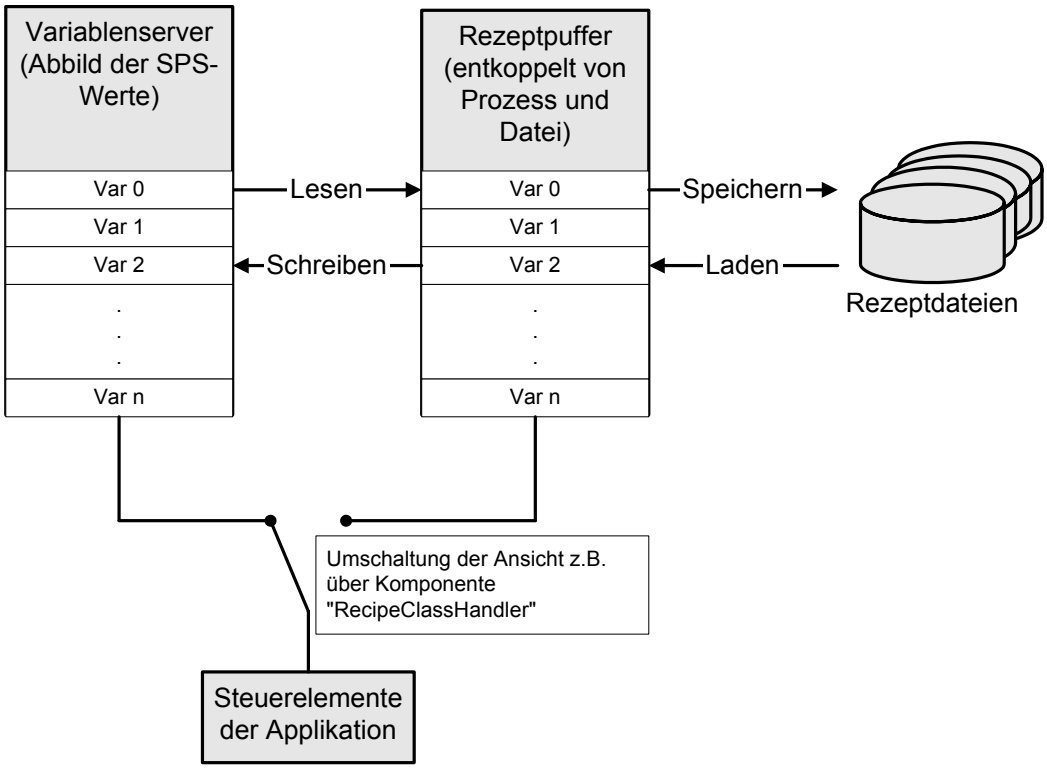
Die laufende Applikation zeigt den in der Trenddefinition angegebenen Prozesswert als Kurve.

2.2.9 Rezepte

Visualisierungsrezepte beschreiben Sätze von Prozesswerten. Diese legen üblicherweise gemeinsam die Einstellungen einer Maschine zur Fertigung eines Produktes aus einer Produktpalette fest. Über das Schreiben von Rezepten zur SPS kann damit zwischen den einzelnen zu fertigenden Produkten umgeschaltet werden.

Zur Entwicklungszeit wird über Rezeptdefinitionen festgelegt, welche Variablen die Funktion gemeinsam beschreiben, welche Variablen also gemeinsam gespeichert und transferiert werden.

Zur Laufzeit werden die Prozesswertsätze in sogenannten Rezeptdateien abgelegt. Zwischen dem Prozessabbild und den Rezeptdateien liegt der Rezeptpuffer. Über diesen Puffer können Rezeptwerte in der Applikation geändert werden, ohne dabei SPS-Daten oder Rezeptwertdaten zu beeinflussen. Der Rezeptpuffer dient zur Parametrierung bzw. zur Optimierung von Rezepten. Aus der Applikation werden beispielsweise Rezeptdaten aus einer Datei geladen, im Rezeptpuffer an das Produktionsziel angepasst und anschließend zur SPS transferiert. Der umgekehrte Weg ermöglicht die Kontrolle eingestellter Werte, bevor eine neue Rezeptdatei erzeugt wird.



Die vier Transferfunktionen "Laden", "Speichern", "Lesen" und "Schreiben" steuern die einzelnen Übertragungsschritte.

Typische Eingabesteuerelemente, wie z.B. das 'VarIn', unterstützen sowohl die Ansicht auf die Prozesswerte im Variablenkern als auch im Rezeptpuffer. Sollen also diese Werte in der Applikation zum Bearbeiten verfügbar gemacht werden, ist die Applikation nicht um neue Steuerelemente zu erweitern. Vielmehr ist die an eine Rezeptdefinition zentral gekoppelte Umschaltung der Ansicht (für alle in der Definition verwendeten Variablen) zu implementieren.

Weitere Funktionen, wie die FDA-konforme Erfassung von Rezeptwertänderungen oder das Speichern von Produktionskommentaren sind einfach parametrierbar.

Die typischen Aufgaben für den Entwickler im Rezeptsystem sind:

Festlegung der Rezeptdefinitionen	Hier wird festgelegt, welche Variablen das Rezept beschreiben. Um allen Irrtümern vorzubeugen: Es geht nicht um die Variablenwerte. Diese werden erst zur Laufzeit festgelegt.
Designanpassung: Editieren der Rezeptwerte	Grundsätzlich können die bestehenden Steuerelemente verwendet werden, die auf die Variablen im Prozess gebunden sind.
Designanpassung: Administration der Rezeptdateien	Typische Aufgaben, wie das Laden oder Speichern von Rezeptwerten aus Dateien, erfordern meist eine Auflistung der bestehenden Rezeptdateien.

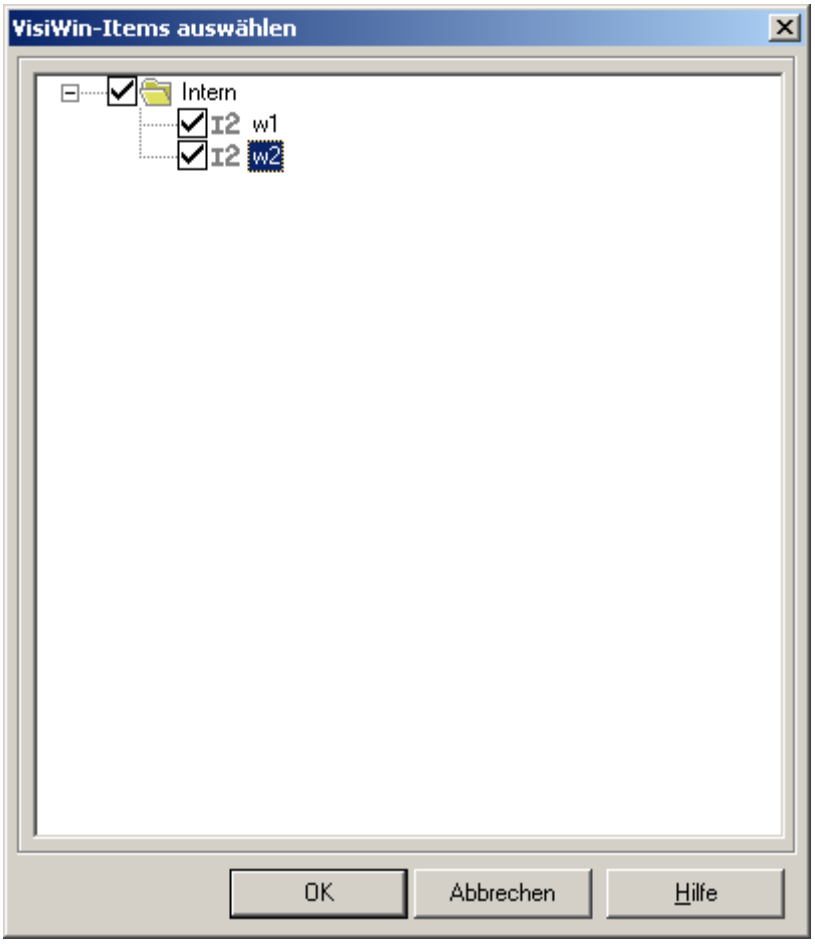
Das folgende Beispiel veranschaulicht die o.g. Schritte anhand eines Minimalrezeptes aus zwei Variablen.

Im Projektexplorer ist der Knoten "Variablen" zu expandieren. Über den Knoten "Kanäle→Intern" ist der Tabelleneditor zu öffnen. Hier sind folgende Variablen zu definieren:

Name	Datentyp
w1	VT_I2
w2	VT_I2

Im Projektexplorer ist der Knoten "Rezepte→Rezeptklassen" zu markieren. Über das Kontextmenü des Knotens ist eine neue Rezeptdefinition zu erzeugen. Der Name der Definition ist in der VisiWinNET-Eigenschaftenseite auf "Rez" einzustellen.

Im Tabelleneditor der neuen Rezeptdefinition ist über den Kontextmenüeintrag "Neu" der Variablenauswahldialog zu öffnen.

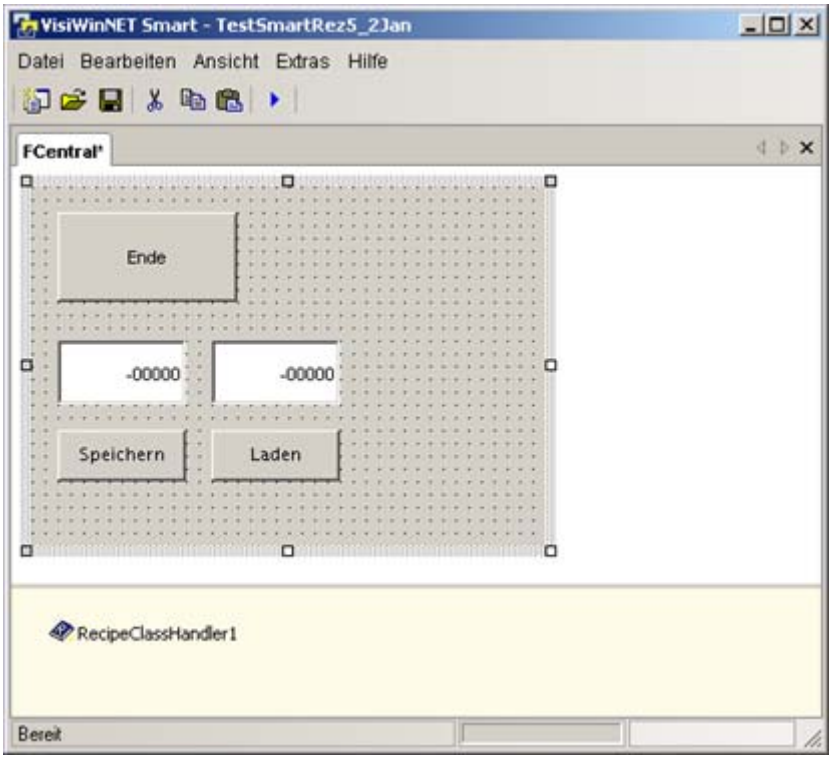


Hier werden die Variablen festgelegt, die zur Rezeptdefinition gehören.

Die beiden internen Variablen sind zu markieren. Der Dialog ist über die OK-Taste zu verlassen.

Die markierten Variablen werden nun als Elemente des Rezeptes im Tabelleneditor aufgeführt.

Auf einem Formular sind zwei 'VarIn', zwei 'CommandButton' und eine 'RecipeClassHandler'-Komponente zu platzieren (Siehe Grafik):



Nachdem die RecipeClassHandler-Komponente auf dem Formular platziert wurde, rutscht sie automatisch in das sogenannte "Komponentenfach" unterhalb der Designansicht des Formulars.

Folgende Eigenschaftenwerte sind hier zu ändern:

Für VarIn1

Eigenschaft	Wert
VWItem	w1

Für VarIn2

Eigenschaft	Wert
VWItem	w2

Für CommandButton1

Eigenschaft	Wert
Text	Speichern
Events	Das Click-Ereignis ist mit dem Aufruf der 'GetRecipe'-Funktion der 'RecipeClassHandler'-Komponente zu verknüpfen. Über die ...-Schaltfläche in der 'Events'-Eigenschaft öffnet sich der Dialog 'Ereignisverarbeitung'. Hier ist das Click-Ereignis auszuwählen und die "Hinzufügen"-Schaltfläche zu drücken. Auf dem nun sichtbaren Funktionsauswahldialog ist die o.g. Funktion auszuwählen. Nach dem Schließen des Funktionsauswahldialoges ist sicherzustellen, dass die Funktion in der Funktionenliste erscheint und ihr Parameter 'Provider' auf die 'RecipeClassHandler'-Komponente des Formulars verweist.

Für CommandButton2

Eigenschaft	Wert
Text	Laden
Events	Das Click-Ereignis ist mit dem Aufruf der 'ShowLoadRecipeDialog'-Funktion der 'RecipeClassHandler'-Komponente zu verknüpfen (s.o.).

Für RecipeClassHandler1

Eigenschaft	Wert
RecipeClass	Rez (Der Name der Rezeptdefinition aus der Projektdatenbank)
Events	Folgende Ereignisse werden hier verwendet: • Das 'GetDoneSucceeded'-Ereignis ist mit dem Aufruf der 'ShowSaveRecipeDialog'-Funktion der 'RecipeClassHandler'-Komponente zu verknüpfen. • Das 'LoadDoneSucceeded'-Ereignis ist mit dem Aufruf der 'SetRecipe'-Funktion der 'RecipeClassHandler'-Komponente zu verknüpfen.

Die so parametrierten Ereignisse bewirken beim Speichern folgende Abläufe:

- Das Klick-Ereignis triggert die Funktion 'GetRecipe' an. Dadurch werden die Werte der als Rezeptelemente festgelegten Variablen in den Rezeptpuffer geladen.
- Den Abschluss des Einlesens meldet der RecipeClassHandler im 'GetDoneSucceeded'-Ereignis. Dieses wiederum ist mit der 'ShowSaveRecipeDialog'-Funktion. Zur Laufzeit wird hierüber das Speichern der Rezeptdatei geregelt, wobei vor dem Speichern ein Dialog zur Eingabe des Namens und der Beschreibung angezeigt wird.

Der umgekehrte Weg wird über die Laden-Schaltfläche ausgelöst. Zuerst wird über einen Auswahldialog eine Rezeptdatei in den Rezeptpuffer geladen und anschließend von dort aus in den Prozess transferiert.

Dieses Beispiel verwendet den Rezeptpuffer praktisch nicht. In vielen Anwendungen mag es ausreichen, wenn Rezepte direkt im Prozess erstellt und optimiert werden.

Die Verwendung des Rezeptpuffers wird aber dann sinnvoll, wenn Produktionseinstellungen über die Applikation vorzunehmen sind, die während des Einstellens die Werte im Prozess nicht verändern dürfen. Erst nach Abschluss der Einstellungen können die Daten dann gemeinsam in den Prozess übertragen werden.

Im Projekt "VWNRecipeDemo" wird beispielhaft die Verwendung des Rezeptpuffers gezeigt.

2.2.10 Benutzerverwaltung

Die VisiWinNET Benutzerverwaltung dient zur Vergabe von personenbezogenen Berechtigungen innerhalb der Visualisierungsapplikation. Eingabesteuerelemente können gesperrt, der Zugang zu Informationen über Ausgabesteuerelemente kann verweigert werden.

VisiWinNET unterstützt zwei Verwaltungssysteme für Benutzer:

ebenenorientierte Benutzerverwaltung

Benutzer und Steuerelemente können einer von 999 Ebenen zugeordnet werden. Ist zur Laufzeit die Ebene eines Steuerelementes höher als die des gerade angemeldeten Benutzers, so wird die Bedienung des Steuerelementes abgelehnt.

rechteorientierte Benutzerverwaltung

Den Steuerelementen in der Applikation wird ein Recht zugeordnet, das der Benutzer besitzen muss, um das Steuerelement bedienen zu können (z.B. "Rezept bearbeiten"). Anschließend werden den Benutzern die Rechte zugeteilt, die sie ausführen dürfen.

Dies ist insbesondere in einem Client-Server-System sinnvoll, wenn mehrere Mitarbeiterhierarchien (Gruppen von Mitarbeitern, die unabhängig voneinander in Berechtigungsebenen aufgeteilt sind) im Netzwerk arbeiten. Über den Vergleich der Rechte von Benutzern und Steuerelementen wird über die Freigabe entschieden.

Die Auswahl des Systems erfolgt im Projektexplorer unter dem Knoten "Benutzerverwaltung→Konfiguration". Nach dem Markieren dieses Knotens ist der Parameter "System" in der VisiWinNET-Eigenschaftenseite einzustellen.

Die Verwaltung von Benutzern wird über drei Definitionen in der Projektdatenbank vorgenommen:

Benutzergruppe

Fasst Benutzer zu einer Gruppe zusammen und enthält die charakteristischen Parameter für alle enthaltenen Benutzer. Über die Benutzergruppe werden die Bezüge auf die zur Bedienung der Applikation nötigen Rechte verwaltet.

Benutzer

Enthält die Zugangsinformationen (Anmeldename und Passwort) und den Aktivierungsstatus eines Benutzers.

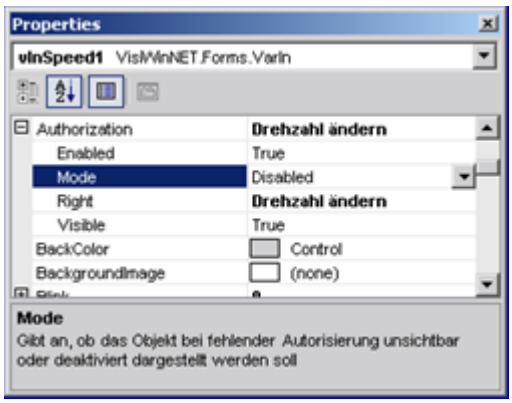
Recht

Rechte definieren die Information, die den Steuerelementen der Applikation zugeordnet werden. Über die "Authorization"-Eigenschaft wird einem Steuerelement ein Recht zugeordnet. Ein angemeldeter Benutzer muss in seiner Benutzergruppendefinition einen Verweis auf dieses Recht haben, damit er das Steurelement bedienen darf.

Rechte werden nur in der rechteorientierten Benutzerverwaltung verwendet. In der ebenenorientierten Benutzerverwaltung erwrtet die "Authorization"-Eigenschaft der Steuerelemente eine Zahl.

Alle drei Definitionstypen können zur Entwicklungszeit über den Benutzerverwaltungseditor projektiert werden.

Alle VisiWinNET-Steuerelemente beinhalten eine 'Authorization'-Eigenschaft. In dieser Eigenschaft wird der Name des Rechtes festgelegt, durch das ein Steuerelement zur Laufzeit bedienbar sein soll. Außerdem wird spezifiziert, wie sich das Steuerelement bei einer fehlenden Berechtigung verhält. Wird die Eigenschaft im Eigenschaftenfenster expandiert, kann in der darunterliegenden 'Mode'-Einstellung zwischen 'disabled' (deaktiviert) und 'Invisible' (unsichtbar) gewählt werden.



Zur Laufzeit stellt VisiWinNET spezielle Dialoge zur Verwaltung der Benutzerinformationen zur Verfügung. Ein typisches Beispiel ist der Anmeldedialog, der als Eingaben den Benutzernamen und ein Passwort verlangt. Über die hier vorgenommene Anmeldung werden die Steuerelemente automatisch je nach vergebener Berechtigung aktiviert oder deaktiviert. Der Aufruf der Benutzerverwaltungsdialoge wird durch die Komponente "UserManagementDialogs" unterstützt.

Das folgende Beispiel zeigt die Arbeitsschritte zur Implementierung einer rechteorientierten Benutzerverwaltung. Beispielhaft wird gezeigt, wie die Verknüpfung der einzelnen Bestandteile auszuführen ist und wie zur Laufzeit Benutzerdaten angepasst werden können. Enthalten sind ein Administrator sowie zwei Benutzer mit unterschiedlichen Rechten. Mithilfe der 'Authorization'-Eigenschaft werden die entsprechenden Freigaben in den Steuerelementen verknüpft.

Außerdem beinhaltet das Beispiel den Aufruf des Anmeldedialoges und des Administrationsdialoges zur Pflege von Benutzerdaten.

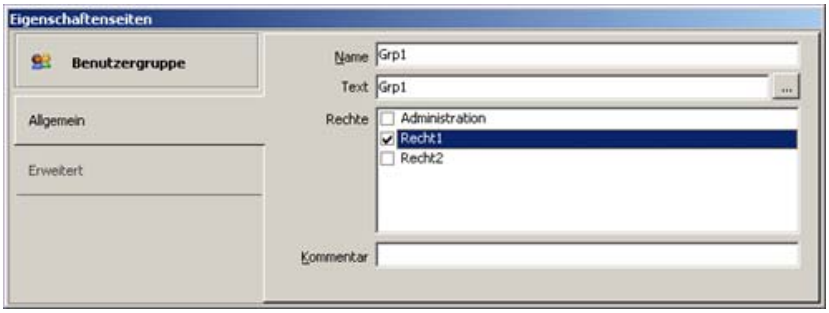
Der Knoten "Benutzerverwaltung→Konfiguration" ist zu markieren. In der VisiWinNET-Eigenschaftenseite ist die Einstellung "System" auf "rechteorientiert" zu setzen.

Im Projektexplorer wird der Knoten "Rechte" eingeblendet.

Der "Rechte"-Knoten ist zu markieren. Im Tabelleneditor sind drei Rechte mit den Namen "Recht1", "Recht2" und "Administration" einzutragen.

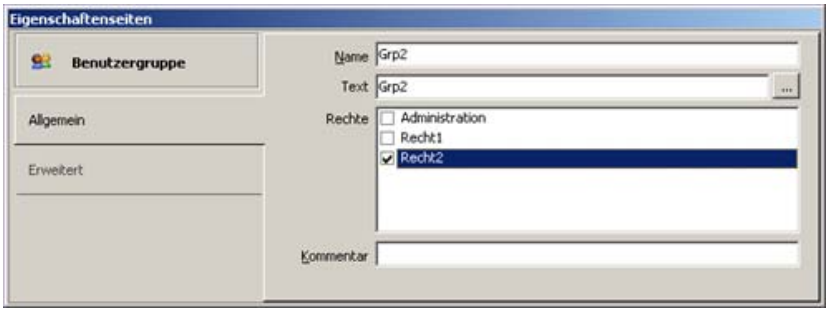
Anschließend ist zu dem Knoten "Benutzergruppen" zu wechseln. Hier sind drei Gruppen anzulegen. Die Parameter sind wie folgt einzustellen:

1. Gruppe



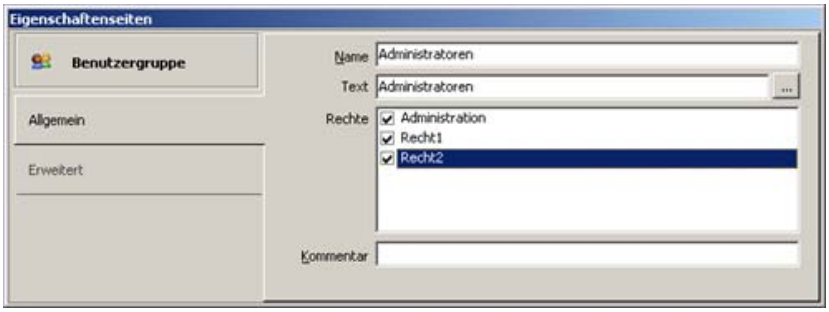
Hier ist der Name anzupassen und die Verknüpfung zu "Recht1" herzustellen.

2. Gruppe



Analog zur ersten Gruppe ist auch hier der Name und die Verknüpfung, wie oben gezeigt, anzupassen.

3.Gruppe



Administratoren dürfen etwas mehr, in diesem Fall alles. Also sind alle Rechte zu verknüpfen.

Die Zuordnung zu Rechten erfolgt über die Benutzergruppe. Das erscheint zwar auf den ersten Blick etwas kompliziert, hilft aber in der realen Welt: Jede Benutzergruppe kann beliebig viele Benutzer aufnehmen. Für alle enthaltenen Benutzer gemeinsam sind die Rechte in der Gruppe einstellbar, also gemeinsam anzupassen.

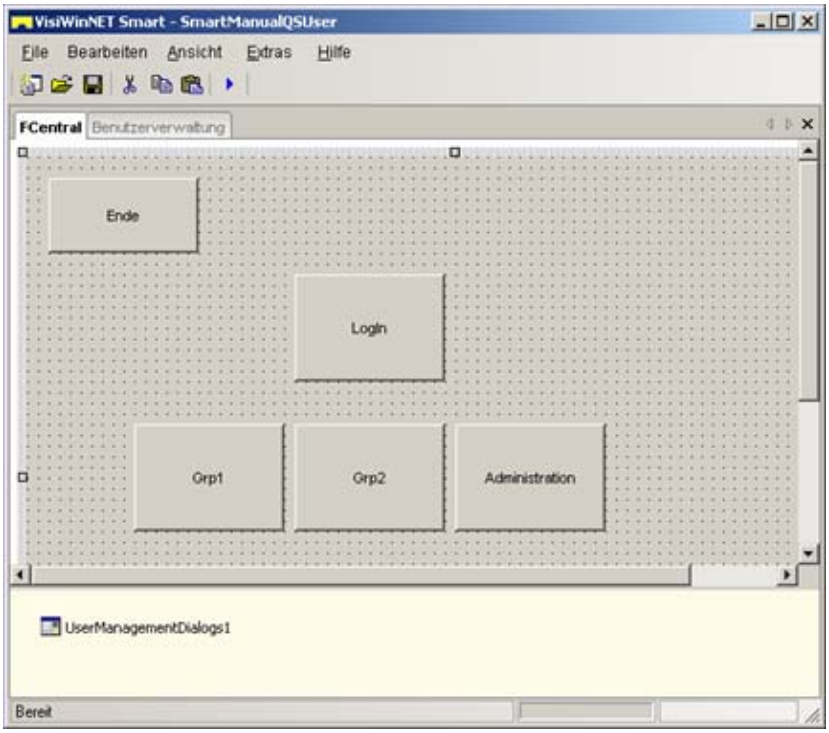
In jeder der drei Benutzergruppen ist im Tabelleneditor ein Benutzer anzulegen. Dabei sind folgende Parameter anzugeben.

Parameter	Wert
Anmeldename	<ein Name>
Passwort	<Ein Passwort das in den Initialeinstellungen mindesten vier Zeichen enthalten muss>
Status	aktiviert

Anmeldename und Passwort werden später als Anmeldeinformationen verwendet. Der Status erlaubt verschiedene Einstellungen, die darüber entscheiden, ob der Benutzer sich anmelden darf und ob er das vom Projekteur vergebene Passwort beim ersten Anmelden ändern muss.

Merken Sie sich die von ihnen vergebenen Anmeldeinformationen für den weiteren Verlauf des Beispiels!

In der Designansicht sind eine 'UserManagementDialogs'-Komponente und vier Schaltflächen vom Typ 'CommandButton' auf einem Formular zu platzieren.



Folgende Eigenschaften sind hier anzupassen:

Für Schaltfläche 1

Text LogOn

Events Das Click-Ereignis ist mit dem Aufruf der 'ShowLogOnDialog'-Funktion der 'UserManagementDialogs'-Komponente zu verknüpfen. Über die ...-Schaltfläche öffnet sich ein Dialog. Hier ist das Click-Ereignis auszuwählen und die "Hinzufügen"-Schaltfläche zu drücken. Auf dem nun sichtbaren Funktionsauswahldialog ist die o.g. Funktion auszuwählen. Nach dem Schließen des Funktionsauswahldialoges ist sicherzustellen, dass die Funktion in der Funktionenliste erscheint und ihr Parameter 'Provider' auf die 'UserManagementDialogs'-Komponente des Formulars verweist.

Für Schaltfläche 2

Text Grp1

Authorization Recht1

Für Schaltfläche 3

Text Grp2

Authorization Recht2

Für Schaltfläche 4

Text Administration

Authorization Administration

Events Das Click-Ereignis ist mit dem Aufruf der 'ShowUserDialog'-Funktion der 'UserManagementDialogs'-Komponente zu verknüpfen (s.o.).

Die Text-Eigenschaft dient hier direkt zur Anzeige der Schaltflächenfunktion.

Die 'Authorization'-Eigenschaft legt jeweils fest, wann die Schaltfläche aktiviert sein soll: Ist ein Benutzer angemeldet und ist seine Benutzergruppe mit dem hier angegebene Recht verknüpft, wird die Schaltfläche aktiviert.

Die Einstellung der 'Events'-Eigenschaft dient zum Aufruf der Dialoge "Benutzer anmelden" und "Benutzer administrieren".

Das Projekt ist zu starten

Nach dem Start ist die Schaltfläche "LogOn" freigeschaltet. Hier wurde keine Einstellung in der 'Authorization'-Eigenschaft vorgenommen. Ein Klick auf die Schaltfläche öffnet den Dialog zum Anmelden eines Benutzers.

Die anderen drei Schaltflächen werden durch die Anmeldung der entsprechenden Benutzer freigeschaltet. Die beiden Schaltflächen "Grp1" und "Grp2" sind funktionslos. Sie werden jeweils über die Benutzer aus den Gruppen "Grp1" und "Administration" bzw. "Grp2" und "Administration" freigeschaltet. Werden so parametrierte Schaltflächen dazu verwendet, zu weiteren Formularen zu wechseln, können ganze Applikationsteile in Abhängigkeit vom aktuellen Benutzer freigegeben werden.

Die Schaltfläche "Administration" steht nur Benutzern aus der gleichnamigen Benutzergruppe zur Verfügung. Ein Klick auf die Schaltfläche öffnet den Benutzerverwaltungsdialog. Hier können neue Benutzer angelegt und bestehende Benutzer verwaltet (deaktiviert, gelöscht oder in den Einstellungen geändert) werden.

Die zur Laufzeit geänderten Benutzerdaten werden nicht in der Projektdatenbank gespeichert. Stattdessen erzeigt das System im Projektverzeichnis eine Datei mit der Endung "rtn".

2.2.11 Protokollierung

Dieses Kapitel befindet sich in der Konstruktionsphase

3 Projekte unter Windows CE

3.1 Projektübertragung aus Visual Studio zum CE-Gerät

Visualisierungsapplikationen, die auf einem Windows CE-Gerät laufen sollen, werden auf einem Standard-PC entwickelt. Von dort aus müssen zur korrekten Funktion alle nötigen Dateien auf das CE-Gerät kopiert werden. Visual Studio enthält Funktionen, die die Verbindung zum Zielgerät konfigurieren, herstellen und auch die Übertragung der ausführbaren Projektdatei, sowie aller Referenzen auf das Zielsystem übernehmen. Mithilfe von Visual Studio lassen sich Projekte auf dem CE-Gerät auch testen (debuggen).

Aufgrund der erweiterten Projektstruktur von VisiWinNET-Visualisierungsapplikationen sind diese Funktionen jedoch nicht ausreichend. Daher erweitert VisiWinNET die Funktionen vom Visual Studio zur Übertragung von Projekten zu einem Windows CE-Gerät.

Im Folgenden wird als Leitfaden beschrieben, wie diese erweiterten Funktionen vom Entwickler zu bedienen sind.

Vorbereitung der Verbindung

RemoteAccessManager auf CE-Gerät kopieren und starten

Die Verbindung zum Zielsystem wird über das Hilfsprogramm "VisiWinNET RemoteAccessManager" hergestellt. Diese Programm ist Teil der VisiWinNET Installation und muss ggf. auf das Windows CE Gerät kopiert werden.

Die entsprechende Datei befindet sich im VisiWin Installationsverzeichnis unter:

<x>:\VisiWin\VisiWinNET\Compact\RemoteAccessManager\

Die Datei kann z.B. per Windows – Netzwerk über eine Freigabe des Entwicklungsrechners kopiert werden. Unter Windows CE öffnet man dazu das "Start" Menü und wählt den Punkt "Run". Danach gibt man den Rechnernamen in der Form \\MeinPC ein. "MeinPC" steht hier für den Netzwerknamen des Entwicklungsrechners. Nach der Ausführung öffnet sich dann entweder eine Ansicht mit den Freigaben des PCs, oder es wird vorher die Eingabe von Benutzername und Passwort angefordert.

Damit das Programm dauerhaft zur Verfügung steht, ist es zweckmäßig, diese Datei auf der Compact Flash des CE Gerätes abzulegen.

Ist das Tool erfolgreich gespeichert, kann es über ein Doppelklick auf dem CE Gerät gestartet werden. Der VisiWinNET RemoteAccessManager zeigt seine Verbindungsbereitschaft nun mit der folgenden Meldung an:

Warte an "IP-Adresse" auf Verbindung.

"IP-Adresse" steht hier als Platzhalter für die lokale IP-Adresse des Windows CE Gerätes.

Das Gerät ist nun bereit, die Verbindung mit der Entwicklungsumgebung aufzunehmen

Gerät in Visual Studio konfigurieren

Folgende Schritte sind auf dem Entwicklungsrechner in Visual Studio auszuführen:

- Öffnen des Optionendialoges von Visual Studio über den Eintrag Extras→Optionen
- Auswahl des Gerätekonfigurationknotens "Gerätetools→Geräte" in der Baumansicht links
- Duplizieren des Eintrages "Windows CE Device" über die Funktionsschaltfläche "Speichern unter". Geben Sie hier einen Namen mit Wiedererkennungswert (z.B. "VWNDevice")an.
- Öffnen des Eigenschaftendialoges des neuen Gerätes über die Funktionsschaltfläche "Eigenschaften"
- Angabe eines vorhandenen Standardausgabeortes (z.B. "Root Folder\Storage", die Angabe richtet sich nach dem Laufwerksnamen auf dem CE-Gerät)
- Öffnen der "TCP Transport Konfiguration". Auswahl der Option "Spezielle IP-Adresse verwenden" und Angabe der IP-Adresse des CE-Gerätes, zu dem das Projekt übertragen werden soll (wird im RemoteAccessManager auf dem CE-Gerät angezeigt).
- Schließen der Dialoge "TCP Transport Konfiguration" und "Geräteeinstellungen"

Optional kann das neue Gerät als Standardgerät auf dem Optionendialog festgelegt werden.

Die so vorgenommenen Einstellungen werden von Visual Studio verwendet. Die erweiterte Übertragungsfunktion von VisiWinNET wird im folgenden Schritt bei der Geräteverbindung konfiguriert.

Verbindungsherstellung

Zur Verbindungsherstellung ist sicherzustellen, dass das im vorherigen Schritt konfigurierte Gerät von Visual Studio verwendet wird. Dazu ist über den Menüeintrag "Projekt→<Projektname> Eigenschaften" der Eigenschaftendialog des Projektes zu öffnen. Auf der Karteikarte "Geräte" befindet sich eine Kombinationsliste "Zielgerät" mit den konfigurierten Geräten. Hier ist das entsprechende Gerät zu wählen.

Beim Start/Übertragen des Projektes wird der von VisiWinNET zur Verfügung gestellte zusätzliche Verbindungsdialog angezeigt.

Hier werden noch einmal die Verbindungsparameter aufgeführt:

Die hier angegebene IP-Adresse stammt aus der Gerätekonfiguration, die vom Visual Studio verwaltet wird. Eine Änderung hat nur für die kommende Verbindung Auswirkungen. Die zusätzliche Angabe eines Ports auf der Karteikarte "Erweitert" wird unabhängig von Visual Studio verwaltet.

Die folgende Verbindung wird in zwei Schritten ausgeführt:

- VisiWinNET-Aufgaben: Übertragung der spezifischen Projektdateien (z.B. Projektdatenbank, Sprachdateien, Smart-Formulare, ...) und der Verbindungsdateien für Visual Studio
- Visual Studio: Übertragung der Referenzen und der ausführbaren EXE. Ggf. Start der Applikation

Zusätzliche VisiWin Funktionen für Geräteverbindungen

VisiWinNET blendet in das Visual Studio eine Werkzeugleiste ein.

Die Schaltflächen der Werkzeugleiste beinhalten die typischen Verbindungsaufgaben:

	Projekt starten. Dabei wird die gesamte Projektübertragung und der Projektstart alleine von VisiWinNET übernommen. Das Debuggen mit Visual Studio funktioniert in dieser Konstellation nicht.
	Projekt stoppen. Gegenüber der gleichnamigen Visual Studio-Funktion wird das Projekt ordnungsgemäß terminiert. Alle geladenen Projektteile (OPC-Server; Applikation, Treiber) werden über '*ExitApplication' in der erforderlichen Reihenfolge beendet.
	Projekt übertragen. Überträgt das Projekt und alle erforderlichen Dateien mit den angegebenen Einstellungen, ohne das Projekt zu starten.
	Verbindung herstellen. Stellt die Verbindung mit den angegebenen Einstellungen her, ohne dass ein Datentransfer stattfindet.
	Verbindung beenden. Schließt die Verbindung
	Verbindung konfigurieren. Öffnet den Dialog zum Einstellen der Verbindungseigenschaften.

4 Weiterführende Referenzen

Dieses Kapitel stellt weitere Hilfestellungen und Informationsquellen vor, die im Lieferumfang des Produktes vorhanden sind. Im Wesentlichen teilen sich diese Informationen in folgende Kategorien:

Hilfe zu den Systemen und Funktionen von VisiWinNET	Beinhaltet die Onlinehilfe zu den VisiWinNET-spezifischen Definitionen, ihren Parametern und der Projektierung
Hilfe zur Klassenbibliothek von VisiWinNET	Enthält die Referenz der Designkomponenten (Steuerelementen und den zugehörigen Klassen und Komponenten)

4.1 Hilfe zu den Systemen und Funktionen von VisiWinNET

Jedes VisiWinNET-Projekt beinhaltet eine Projektdatenbank, in der zur Entwicklungszeit spezielle Definitionen mithilfe von Editoren projektiert werden. Typische Definitionen sind z.B. die in der Applikation verwendeten Variablen, Meldungen oder sprachumschaltbare Texte. Jede Definition beinhaltet einen Satz von Parametern, die das Verhalten zur Laufzeit festlegen. Die Laufzeitkomponenten erzeugen anhand der Definitionen entsprechende Objekte im Arbeitsspeicher, die der Visualisierung zur Verfügung stehen.

Die Hilfe zu den Systemen und Funktionen beinhaltet im wesentlichen:

- Die Bedienung der Editoren, die zum Erzeugen der Definitionen in der Projektdatenbank dienen
- Die Beschreibung der Definitionen und ihrer Parameter

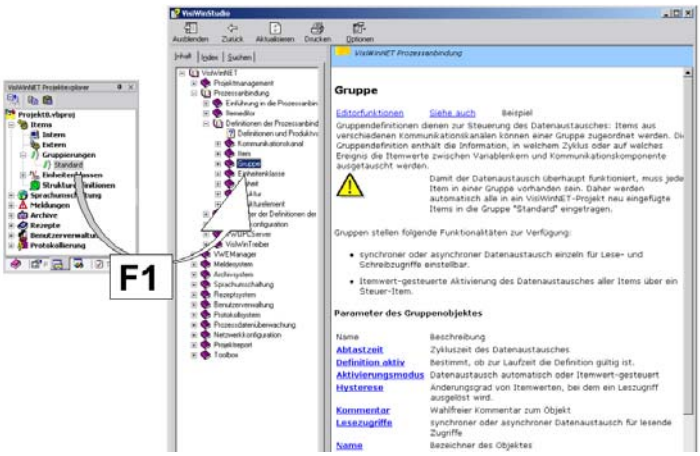
Diese Hilfe ist in Form einer Onlinehilfe hinterlegt.

4.1.1 Aufruf

Der Aufruf dieser Hilfe erfolgt über den Projektexplorer oder in den Editoren:

Projektexplorer

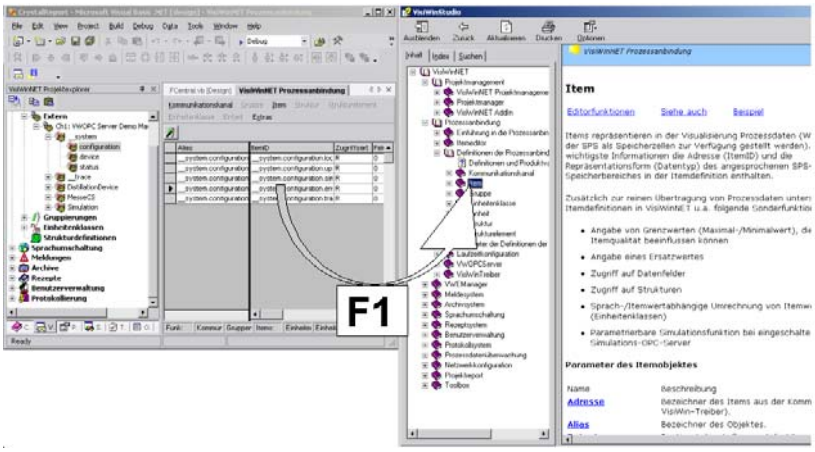
Im Projektexplorer kann eine Definition selektiert werden. Über die Taste "F1" wird die entsprechende Hilfe der selektierten Definition aufgerufen.



Die Hilfe enthält die Beschreibung der Definition und eine Verweisliste auf ihre Parameter. Wird ein Verweis angeklickt, zeigt die Hilfe das entsprechende Thema über den gewählten Parameter an.

Tabelleneditor

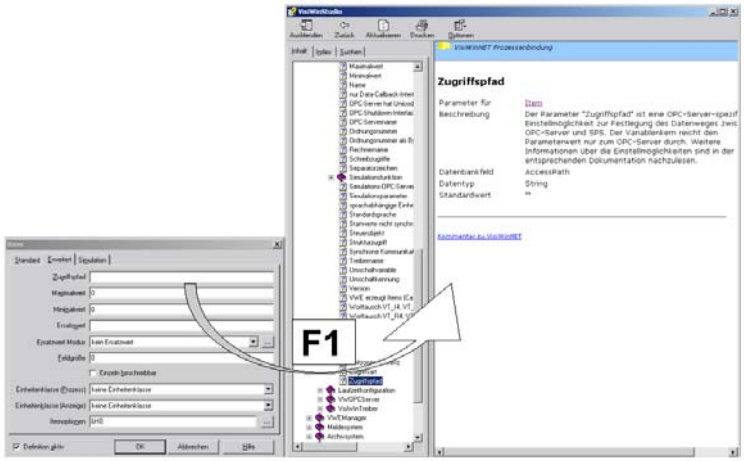
Auch im Tabelleneditor kann die Hilfe der angezeigten Definitionen über "F1" aufgerufen werden.



Dialoge

Die Dialoge der Definitionen ermöglichen den direkten Zugriff auf die Hilfe zu einem Parameter. Wird der Dialog zum Bearbeiten einer Definition geöffnet, kann die Onlinehilfe auf zwei Arten geöffnet werden:

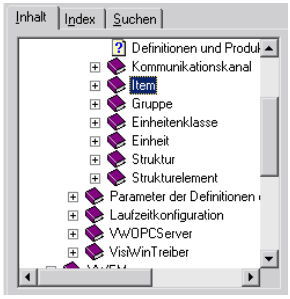
- Über die Hilfe-Schaltfläche wird die entsprechende Hilfe zur Definition angezeigt
- Nach dem Setzen der Eingabemarke auf ein Feld ,das den Wert eines Parameters anzeigt, wird über die "F1"-Taste direkt die Hilfe zum entsprechenden Parameter angezeigt.



4.1.2 Recherche

Hat der Entwickler die Hilfe erst einmal erreicht, kann er weitere Recherchen anstellen. Über die Verweise der aktuell angezeigten Seite können weitere Themen aufgerufen werden, die thematisch mit der aktuellen Seite verknüpft sind. Das Hilfesystem enthält außerdem weitere Funktionen, die bei der Recherche helfen. Der linke Teil des Hilfefenster enthält dazu drei Karteikarten:

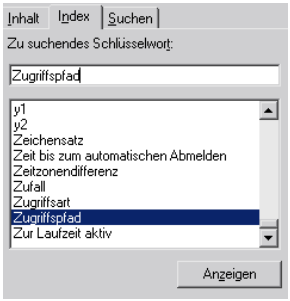
Inhalt



Der Inhaltsbaum bildet eine hierarchische Ansicht des Inhaltsverzeichnisses. Er beinhaltet die Übersicht über alle VisiWinNET-Systeme. Jedes System gliedert sich im wesentlichen in folgende Kapitel:

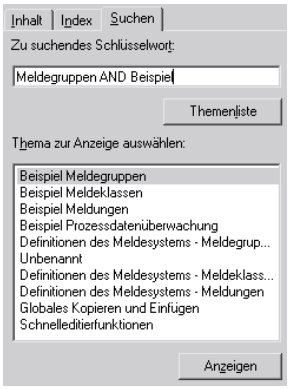
Einführung	Eine Beschreibung des Systems mit Ablaufdiagramm und Vorstellung der einzelnen Bestandteile
Projektierung	Die Bedienreferenz des Editors
Definitionen	Die Beschreibung der Definitionen, die im System verwendet werden können.
Parameter	Die Beschreibung der Parameter, die den Definitionen zur Verfügung stehen.

Index



Der Index enthält eine alphabetisch sortierte Liste aller zur Verfügung stehender Schlagwörter. Hier werden z.B. alle Parameter- und Definitionsnamen aufgeführt. Verwendet wird der Index, wenn der zu suchende Begriff zwar bekannt ist, jedoch der direkte Aufruf aus der Entwicklungsumgebung und der Standort einer Hilfeseite im Inhaltsbaum nicht bekannt sind.

Suchen



Die "Suchen"-Funktion ermöglicht eine Intensivrecherche nach einem Begriff. Nach Angabe des Suchbegriffs führt die Hilfe eine Volltextsuche über alle Hilfeseiten durch. Hilfeseiten, die den Begriff enthalten, werden als Treffer aufgelistet. Neben der Angabe kombinierter Begriffe (über das Schlüsselwort "AND") können auch Platzhalter (Wildcards "*") verwendet werden.

4.2 Hilfe zur Klassenbibliothek von VisiWinNET

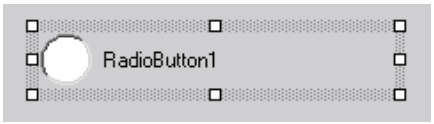
Die Hilfe zur Klassenbibliothek beinhaltet die Beschreibung aller in Visual Studio .NET zur Verfügung stehender "Typen". "Typen" sind hier im wesentlichen Steuerelemente, Komponenten und Klassen, die von den Programmiersprachen VB.NET und C# interpretiert und kompiliert werden können.

4.2.1 Aufruf

Der Aufruf der Hilfe kann über mehrere Wege erfolgen:

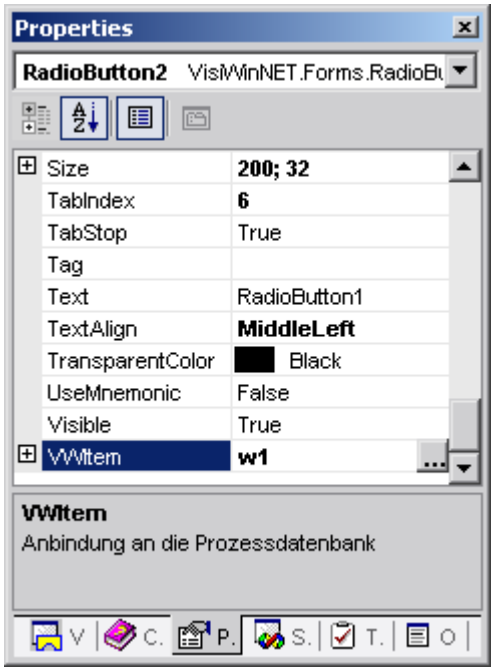
Selektieren eines Steuerelementes auf einem Formular, anschließend Taste "F1"

Die entsprechende Hilfeseite zum Steuerelement wird geöffnet.



Selektieren einer Eigenschaft im Eigenschaftenfenster, anschließend Taste "F1"

Die Hilfe zur entsprechenden Eigenschaft wird geöffnet. Vor der "F1"-Taste wird schon im unteren Fensterbereich eine Kurzbeschreibung zur Eigenschaft angezeigt.



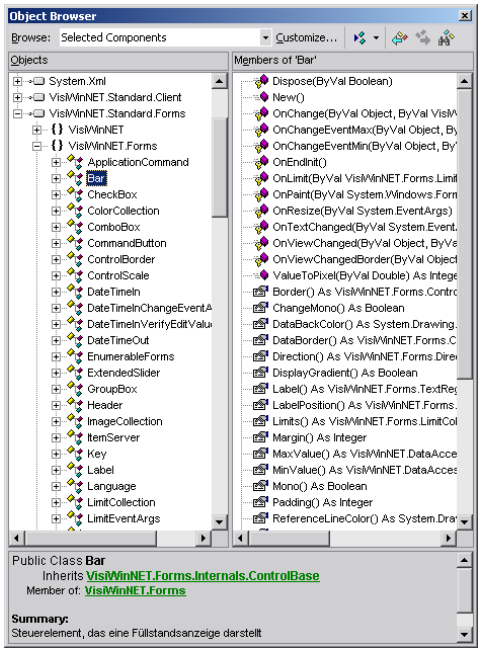
Setzen der Eingabemarke auf ein Wort im Quelltext, anschließend Taste "F1"

Die Hilfe zur entsprechenden Eigenschaft/zum entsprechenden Typ oder Schlüsselwort wird geöffnet.

```
Private Sub CommandButton1_Click(
    RadioButton1.StopEdit(True)
End Sub
```

Selektieren eines Eintrages im Objektkatalog, anschließend Taste "F1"

Der Objektkatalog wird über den Menüeintrag "Ansicht→Objektkatalog" sichtbar geschaltet. Er enthält für alle im Projekt eingebundenen Typen eine Auflistung der zur Verfügung stehenden "Member" (Eigenschaften, Ereignisse, Methoden oder Konstanten). Im unteren Teil des Fensters wird eine Kurzbeschreibung für den selektierten Typ/ das selektierte Member angezeigt. Die "F1"-Taste öffnet die entsprechende Hilfeseite.



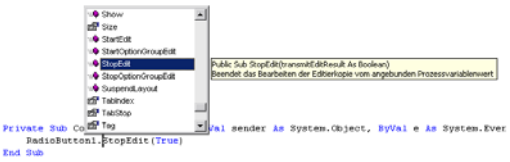
Dynamische Hilfe

Eine weitere Möglichkeit, die Hilfe aufzurufen, besteht über das dynamische Hilfefenster. Dieses Fenster gibt eine kontextbezogene (von den aktuell in der Entwicklungsoberfläche sichbaren/selektierten Elementen abhängige) Liste von Hilfethemen an. Der Inhalt des Fensters springt alleine durch die Selektion in der Entwicklungsumgebung um. Wird also z.B. ein Fenster geöffnet, zeigt das dynamische Hilfefenster verwandte Hilfethemen zu diesem Fenster an.

Ein Klick auf einen Listeneintrag im dynamischen Hilfefenster startet die entsprechende Hilfeseite.

Das dynamische Hilfefenster wird über den Menüeintrag "Hilfe→Dynamische Hilfe" sichtbar geschaltet.

Intellisense

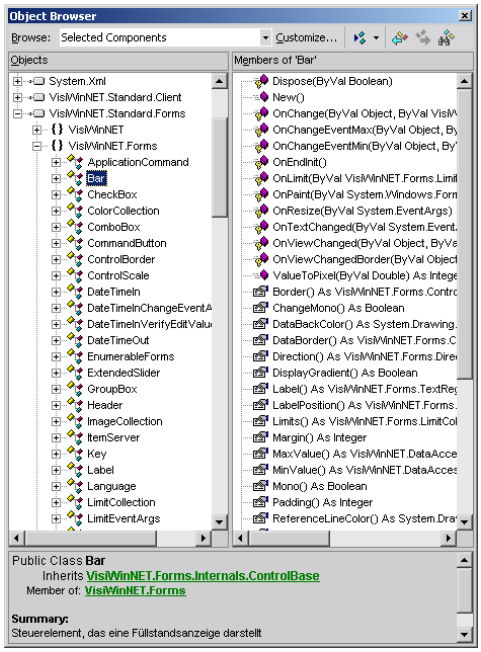


Beim Schreiben von Quellcode unterstützt die Entwicklungsumgebung den Projekteur durch zwei Fenster:

- Durch Setzen des "."-Operator öffnet sich eine Auswahlliste, die die entsprechenden "Member" (Eigenschaften, Ereignisse, Methoden oder Konstanten) des vor dem "."-Operator angegebenen Objektes enthält.
- Über die Navigationstasten (↑↓) kann zwischen den Listeneinträgen gewechselt werden. Dabei wird ein Tooltip angezeigt, der die Kurzbeschreibung des in der Liste selektierten "Member" anzeigt.

4.2.2 Inhalt der Klassenbibliothekshilfe

Der Aufbau der Klassenbibliothekshilfe folgt dem Programmiermodell von .NET. Er ist am besten anhand des Objektkataloges nachzuvollziehen:



VisiWinNET stellt dem Entwickler zwei Assemblies zur Verfügung, deren Typen und Member in der Visualisierungsapplikation verwendet werden können:

Client

Je nach Produktvariante z.B. "VisiWinNET.Standard.Client" oder "VisiWinNET.Compact.Client"

Bildet die programmatische Schnittstelle zu den Systemservern (Variablenkern, Alarmserver, ...).

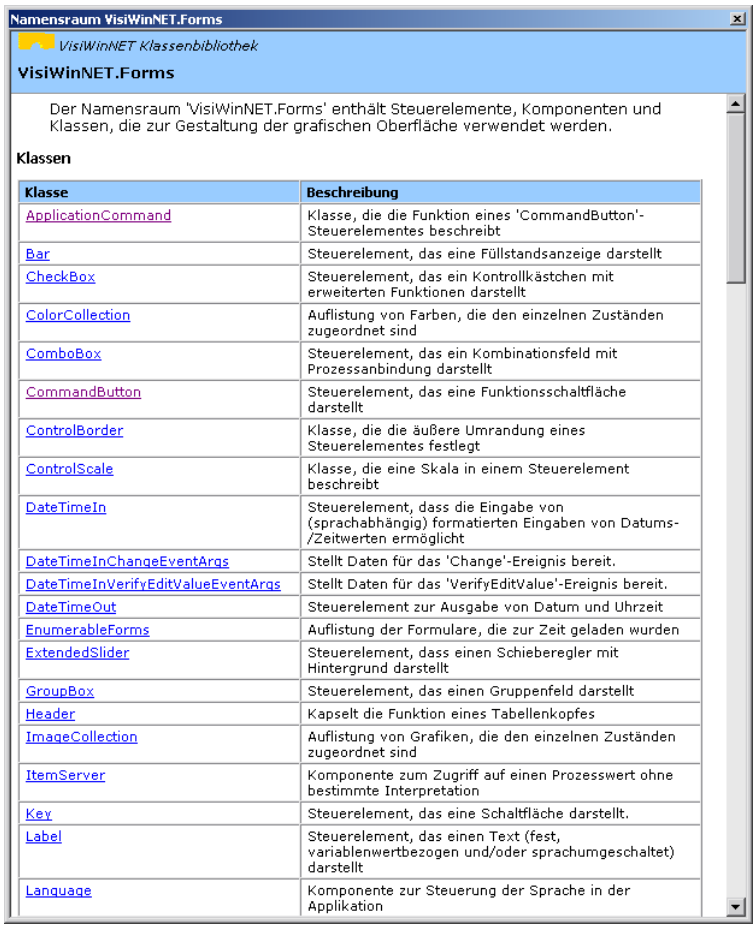
Forms

Je nach Produktvariante z.B. "VisiWinNET.Standard.Forms" oder "VisiWinNET.Compact.Forms"

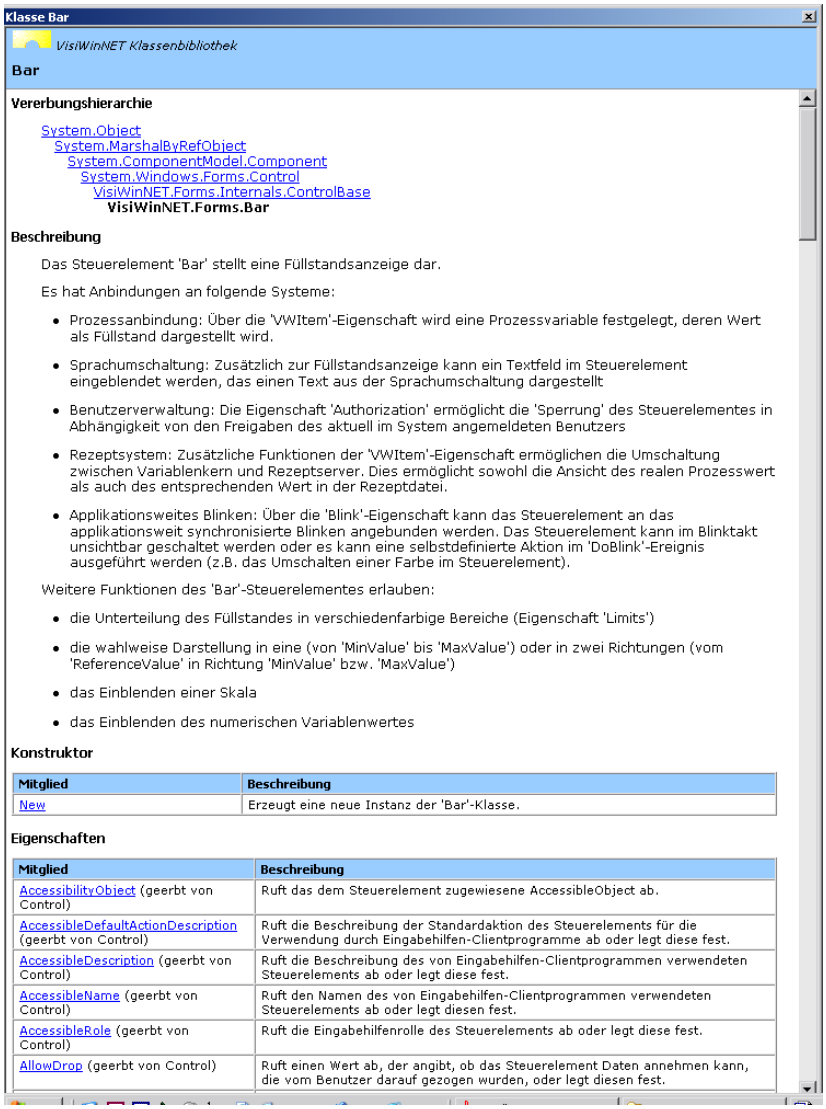
Enthält die funktionalen Elemente und Klassen für die Applikation (unter anderem auch die Steuerelemente). Intern nutzen die hier enthaltenen Typen die Funktionen des Clients.

Die Typen eines Assemblies sind in sogenannte "Namensräume" (Namespaces) unterteilt. Namensräume gliedern die Typen eines Assemblies. Z.B. werden die VisiWinNET-Systeme (Alarm/Archiv/Prozessanbindung) im Client in entsprechende Namensräume (Alarm/Trend/DataAccess) gegliedert.

Der für eine Applikation wichtigste Namensraum ist für den Entwickler "VisiwinNET.Forms". Hier werden die am häufigsten verwendeten Steuerelemente und Klassen aufgelistet. Ein erster grober Überblick wird über die Hilfeseite des Namensraumes möglich (Namensraum im Objektkatalog selektieren, anschließend "F1" drücken.


Namensraum VisiWinNET.Forms
VisiWinNET Klassenbibliothek
VisiWinNET.Forms
Der Namensraum 'VisiWinNET.Forms' enthält Steuerelemente, Komponenten und Klassen, die zur Gestaltung der grafischen Oberfläche verwendet werden.
Klassen
Klasse
Beschreibung
ApplicationCommand
Klasse, die die Funktion eines 'CommandButton'-Steuerelementes beschreibt
Bar
Steuerelement, das eine Füllstandsanzeige darstellt
CheckBox
Steuerelement, das ein Kontrollkästchen mit erweiterten Funktionen darstellt
ColorCollection
Auflistung von Farben, die den einzelnen Zuständen zugeordnet sind
ComboBox
Steuerelement, das ein Kombinationsfeld mit Prozessanbindung darstellt
CommandButton
Steuerelement, das eine Funktionsschaltfläche darstellt
ControlBorder
Klasse, die die äußere Umrandung eines Steuerelementes festlegt
ControlScale
Klasse, die eine Skala in einem Steuerelement beschreibt
DateTimeIn
Steuerelement, dass die Eingabe von (sprachabhängig) formatierten Eingaben von Datums-/Zeitwerten ermöglicht
DateTimeInChangeEventArgs
Stellt Daten für das 'Change'-Ereignis bereit.
DateTimeInVerifyEditValueEventArgs
Stellt Daten für das 'VerifyEditValue'-Ereignis bereit.
DateTimeOut
Steuerelement zur Ausgabe von Datum und Uhrzeit
EnumerableForms
Auflistung der Formulare, die zur Zeit geladen wurden
ExtendedSlider
Steuerelement, dass einen Schieberegler mit Hintergrund darstellt
GroupBox
Steuerelement, das einen Gruppenfeld darstellt
Header
Kapselt die Funktion eines Tabellenkopfes
ImageCollection
Auflistung von Grafiken, die den einzelnen Zuständen zugeordnet sind
ItemServer
Komponente zum Zugriff auf einen Prozesswert ohne bestimmte Interpretation
Key
Steuerelement, das eine Schaltfläche darstellt.
Label
Steuerelement, das einen Text (fest, variablenwertbezogen und/oder sprachumgeschaltet) darstellt
Language
Komponente zur Steuerung der Sprache in der Applikation

Diese Hilfeseite listet stichpunktartig alle Typen des selektierten Namensraum auf. Die Verweise in der linken Tabellenspalte führen jeweils auf die Detailbeschreibungen.


Klasse Bar
VisiWinNET Klassenbibliothek
Bar
Vererbungshierarchie
System.Object
System.MarshalByRefObject
System.ComponentModel.Component
System.Windows.Forms.Control
VisiWinNET.Forms.Internals.ControlBase
VisiWinNET.Forms.Bar
Beschreibung
Das Steuerelement 'Bar' stellt eine Füllstandsanzeige dar.
Es hat Anbindungen an folgende Systeme:
Prozessanbindung: Über die 'VWItem'-Eigenschaft wird eine Prozessvariable festgelegt, deren Wert als Füllstand dargestellt wird.
Sprachumschaltung: Zusätzlich zur Füllstandsanzeige kann ein Textfeld im Steuerelement eingeblendet werden, das einen Text aus der Sprachumschaltung dargestellt
Benutzerverwaltung: Die Eigenschaft 'Authorization' ermöglicht die 'Sperrung' des Steuerelementes in Abhängigkeit von den Freigaben des aktuell im System angemeldeten Benutzers
Rezeptsystem: Zusätzliche Funktionen der 'VWItem'-Eigenschaft ermöglichen die Umschaltung zwischen Variablenkern und Rezeptserver. Dies ermöglicht sowohl die Ansicht des realen Prozesswert als auch des entsprechenden Wert in der Rezeptdatei.
Applikationsweites Blinken: Über die 'Blink'-Eigenschaft kann das Steuerelement an das applikationsweit synchronisierte Blinken angebunden werden. Das Steuerelement kann im Blinktakt unsichtbar geschaltet werden oder es kann eine selbstdefinierte Aktion im 'DoBlink'-Ereignis ausgeführt werden (z.B. das Umschalten einer Farbe im Steuerelement).
Weitere Funktionen des 'Bar'-Steuerelementes erlauben:
die Unterteilung des Füllstandes in verschiedenfarbige Bereiche (Eigenschaft 'Limits')
die wahlweise Darstellung in eine (von 'MinValue' bis 'MaxValue') oder in zwei Richtungen (vom 'ReferenceValue' in Richtung 'MinValue' bzw. 'MaxValue')
das Einblenden einer Skala
das Einblenden des numerischen Variablenwertes
Konstruktor
Mitglied
Beschreibung
New
Erzeugt eine neue Instanz der 'Bar'-Klasse.
Eigenschaften
Mitglied
Beschreibung
AccessibilityObject (geerbt von Control)
Ruft das dem Steuerelement zugewiesene AccessibleObject ab.
AccessibleDefaultActionDescription (geerbt von Control)
Ruft die Beschreibung der Standardaktion des Steuerelements für die Verwendung durch Eingabehilfen-Clientprogramme ab oder legt diese fest.
AccessibleDescription (geerbt von Control)
Ruft die Beschreibung des von Eingabehilfen-Clientprogrammen verwendeten Steuerelements ab oder legt diese fest.
AccessibleName (geerbt von Control)
Ruft den Namen des von Eingabehilfen-Clientprogrammen verwendeten Steuerelements ab oder legt diesen fest.
AccessibleRole (geerbt von Control)
Ruft die Eingabehilfenrolle des Steuerelements ab oder legt diese fest.
AllowDrop (geerbt von Control)
Ruft einen Wert ab, der angibt, ob das Steuerelement Daten annehmen kann, die vom Benutzer darauf gezogen wurden, oder legt diesen fest.

Hilfeseiten für "Typen" (Steuerelement, Komponente, Klasse, Aufzählung) sind in folgende Punkte gegliedert:

Vererbungs-hierarchie Darstellung, von welchen Klassen der Typ erbt bzw. welche weiteren Klassen die Eigenschaften des Typen erben.

Beschreibung Beschrieben wird, welche Anwendungsfälle der Typ unterstützt und welche besonderen Funktionen er gegebenenfalls enthält.

Member-Liste Es folgt eine Liste aller Member des Steuerelementes gegliedert in "Konstuktor", "Eigenschaften", "Ereignisse", "Methoden" und "Konstanten". Jedes Member wird mit einer Kurzbeschreibung angegeben. Vererbte Members werden mit einem entsprechenden Kommentar versehen.

Hinweise Danach folgen Bemerkungen, die gegebenenfalls auf Besonderheiten des Typs eingehen.

Beispiel Falls vorhanden, wird ein Beispiel zur Verwendung des Typs angegeben

Siehe auch

Abschließend wird eine Verweisliste angeboten, die den Zugriff auf thematisch zusammenhängende Hilfeseiten ermöglicht. Dies sind:

- Der Namensraum, in dem der Typ liegt
- Member, die den Typen als Parameter oder Rückgabewert verwenden
- Verwandte Themen, deren Verwendung mit dem Typen zusammenhängen

Hilfeseiten für "Member" (Eigenschaften, Ereignisse, Methoden oder Konstanten) sind in folgende Punkte gegliedert:

Syntax

Für die Programmiersprachen C# und VB.NET wird die Syntax aufgezeigt (also wie das Member im Quellcode zu verwenden ist). Dabei werden alle verwendeten Datentypen als Verweise angeboten. Dies ist besonders dann wichtig, wenn z.B. eine Eigenschaft ein Objekt als Datentyp erwartet. Will der Entwickler nach der Lektüre der aktuellen Hilfeseite weitere Informationen zum zu übergebenden Datentypen haben, kann er über den Verweis in der Syntaxbeschreibung direkt zur Hilfe des Datentyps wechseln.

Beschreibung

Es folgt eine kurze Beschreibung zur Funktion.

Hinweise

Falls erforderlich, werden hier besondere Bemerkungen aufgeführt

Beispiel

Falls vorhanden, wird ein Beispiel zur Verwendung angegeben

Siehe auch

Abschließend wird eine Verweisliste angeboten, die den Zugriff auf thematisch zusammenhängende Hilfeseiten ermöglicht. Dies sind:

- Der Namensraum, in dem der Typ liegt
- Der Typ, der das "Member" implementiert
- Verwandte Themen, deren Verwendung mit dem "Member" zusammenhängen

4.3 Handbuch

4.3.1 PDF-Dateien

Die Handbücher liegen in elektronischer Form als Sammlung von pdf-Dateien auf der CD im Verzeichnis "VisiWinNET\Common\Documentation" vor. Die PDF-Dateien sind in Themen gegliedert. Sie können mit Hilfe des Adobe Acrobat Readers betrachtet und ausgedruckt werden. Der Acrobat Reader ist als Installationsversion auf der CD im Verzeichnis "AcrobatReader" vorhanden (Bitte beachten Sie bei der Installation die Adobe-Lizenzvereinbarungen).

Alle Informationen, die im Handbuch enthalten sind, befinden sich auch in der Onlinehilfe. Es ist also nicht zwingend nötig, das gesamte Handbuch mit einem Volumen von knapp 2000 Seiten auszudrucken.